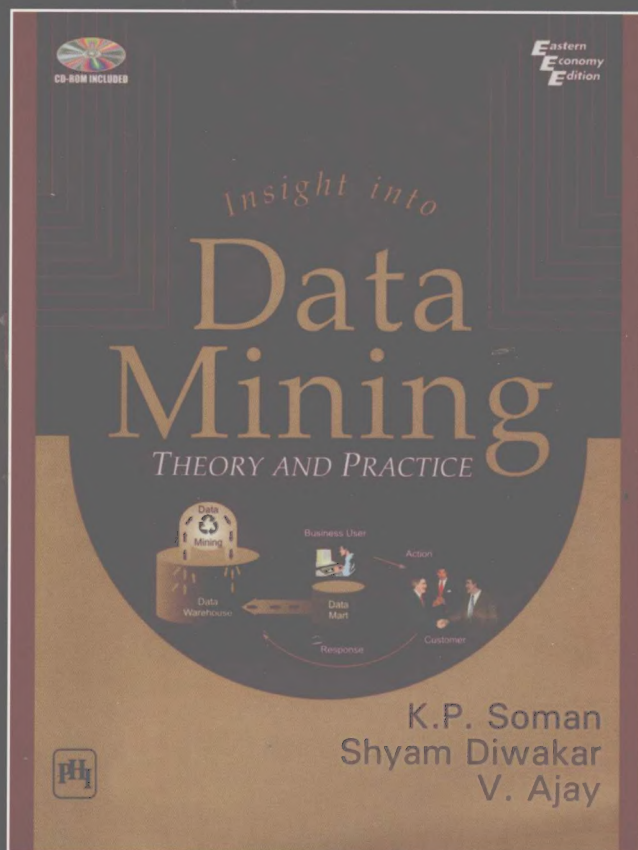


数据挖掘基础教程

(印度) K. P. Soman Shyam Diwakar V. Ajay 著 范明 牛常勇 译



Insight into Data Mining
Theory and Practice



机械工业出版社
China Machine Press

数据挖掘基础教程

数据挖掘是一个新兴的多学科交叉领域，它基于人工智能、机器学习、模式识别、统计学、数据库、可视化等技术，能够从数据库的大量数据中揭示出隐含的、先前未知的并有潜在价值的信息，目前已广泛应用于科学、工程、商业、医学等领域。

本书旨在向读者介绍数据挖掘方法和算法，使读者能够应用这些方法解决现实世界中的问题。本书精心选择了在数据挖掘领域中广泛使用的大部分方法，并辅以简单的例子，因而是学习数据挖掘的理想教材。

本书特色

- 涵盖数据挖掘中数据的预处理、分类、预测、聚类、关联、支持向量机、多维数据可视化等内容，以及用于这些数据挖掘问题的典型算法。
- 许多算法都通过例子解释，并辅以大量图示，有利于初学者理解。
- 介绍如何使用开源软件包Weka和ExcelMiner、GCLUTO工具进行数据挖掘。在学习理论的同时，配合使用这些数据挖掘软件进行实验有利于读者加深对数据挖掘理论和算法的理解。
- 介绍了一些源自UCI机器学习库的数据集，它们已经成为研究算法性能的基准数据集。

附带光盘包括

- 大量数据集。
- 使用Weka和ExcelMiner进行数据挖掘的演示。

投稿热线: (010) 88379604
购书热线: (010) 68995259, 68995264
读者信箱: hzsj@hzbook.com

华章网站 <http://www.hzbook.com>

网上购书: www.china-pub.com

封面设计: 包弼 林杉



上架指导: 计算机 / 数据挖掘

ISBN 978-7-111-25543-7

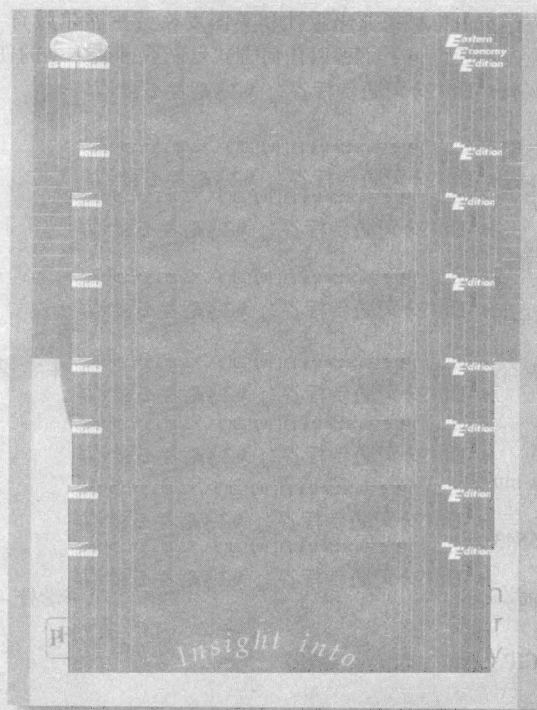


定价: 45.00元 (附光盘)

计 算 机 科 学 丛 书

数据挖掘基础教程

(印度) K. P. Soman Shyam Diwakar V. Ajay 著 范明 牛常勇 译



Insight into Data Mining
Theory and Practice



机械工业出版社
China Machine Press

本书全面介绍数据挖掘的原理、方法和算法。主要内容包括数据挖掘的基本概念、数据挖掘算法的数据类型、输入和输出、决策树、数据挖掘的预处理和后处理、关联规则挖掘、分类和回归算法、支持向量机、聚类分析及多维数据可视化。

本书讲解深入浅出,并辅以大量实例,随书光盘提供了大量数据集以及两种广泛使用的数据挖掘软件——Weka 和 ExcelMiner,便于读者理解数据挖掘知识。

本书适合作为高等院校计算机及相关专业数据挖掘课程的教材,也可供广大技术人员参考。

Insight into Data Mining: Theory and Practice

By K. P. Soman, Shyam Diwakar, V. Ajay.

Copyright © 2006 by Prentice Hall of India.

All rights reserved.

Originally published in India by Prentice Hall of India.

Chinese (in simplified character only) translation rights arranged with Prentice Hall of India through McGraw-Hill Education (Asia).

本书中文简体字翻译版由机械工业出版社和美国麦格劳-希尔教育(亚洲)出版公司合作出版。未经出版者预先书面许可,不得以任何方式复制或抄袭本书的任何部分。

本书封面贴有 McGraw-Hill 公司防伪标签,无标签者不得销售。

版权所有,侵权必究。

本书法律顾问 北京市展达律师事务所

本书版权登记号:图字:01-2007-3810

图书在版编目(CIP)数据

数据挖掘基础教程/(印度)西蒙(Soman, K. P.)等著;范明,牛常勇译. —北京:机械工业出版社,2009.1

(计算机科学丛书)

书名原文:Insight into Data Mining: Theory and Practice

ISBN 978-7-111-25543-7

I. 数… II. ①西… ②范… ③牛… III. 数据采集-高等学校-教材 IV. TP274

中国版本图书馆 CIP 数据核字(2008)第 177403 号

机械工业出版社(北京市西城区百万庄大街 22 号 邮政编码 100037)

责任编辑:朱 劼

北京慧美印刷有限公司印刷

2009 年 1 月第 1 版第 1 次印刷

184mm × 260mm · 19.75 印张

标准书号:ISBN 978-7-111-25543-7

ISBN 978-7-89482-887-3(光盘)

定价:45.00 元(附光盘)

凡购本书,如有倒页、脱页、缺页,由本社发行部调换
本社购书热线:(010)68326294

出版者的话

文艺复兴以降，源远流长的科学精神和逐步形成的学术规范，使西方国家在自然科学的各个领域中取得了垄断性的优势；也正是这样的传统，使美国在信息技术发展的六十多年间名家辈出、独领风骚。在商业化的进程中，美国的产业界与教育界越来越紧密地结合，计算机学科中的许多泰山北斗同时身处科研和教学的最前线，由此而产生的经典科学著作，不仅筹划了研究的范畴，还揭示了学术的源变，既遵循学术规范，又自有学者个性，其价值并不会因年月的流逝而减退。

近年，在全球信息化大潮的推动下，我国的计算机产业发展迅猛，对专业人才的需求日益迫切。这对计算机教育界和出版界都既是机遇，也是挑战；而专业教材的建设在教育战略上显得举足轻重。在我国信息技术发展时间较短的现状下，美国等发达国家在其计算机科学发展的几十年间积淀和发展的经典教材仍有许多值得借鉴之处。因此，引进一批国外优秀计算机教材将对我国计算机教育事业的发展起到积极的推动作用，也是与世界接轨、建设真正的世界一流大学的必由之路。

机械工业出版社华章分社较早意识到“出版要为教育服务”。自1998年开始，华章分社就将工作重点放在了遴选、移译国外优秀教材上。经过多年的不懈努力，我们与Pearson, McGraw-Hill, Elsevier, MIT, John Wiley & Sons, Cengage等世界著名出版公司建立了良好的合作关系，从他们现有的数百种教材中甄选出Andrew S. Tanenbaum, Bjarne Stroustrup, Brain W. Kernighan, Dennis Ritchie, Jim Gray, Alfred V. Aho, John E. Hopcroft, Jeffrey D. Ullman, Abraham Silberschatz, William Stallings, Donald E. Knuth, John L. Hennessy, Larry L. Peterson等大师名家的一批经典作品，以“计算机科学丛书”为总称出版，供读者学习、研究及珍藏。大理石纹理的封面，也正体现了这套丛书的品位和格调。

“计算机科学丛书”的出版工作得到了国内外学者的鼎力襄助，国内的专家不仅提供了中肯的选题指导，还不辞劳苦地担任了翻译和审校的工作；而原书的作者也相当关注其作品在中国的传播，有的还专程为其书的中译本作序。迄今，“计算机科学丛书”已经出版了近两百个品种，这些书籍在读者中树立了良好的口碑，并被许多高校采用为正式教材和参考书籍。其影印版“经典原版书库”作为姊妹篇也被越来越多实施双语教学的学校所采用。

权威的作者、经典的教材、一流的译者、严格的审校、精细的编辑，这些因素使我们的图书有了质量的保证。随着计算机科学与技术专业学科建设的不断完善和教材改革的逐渐深化，教育界对国外计算机教材的需求和应用都将步入一个新的阶段，我们的目标是尽善尽美，而反馈的意见正是我们达到这一终极目标的重要帮助。华章分社欢迎老师和读者对我们的工作提出建议或给予指正，我们的联系方式如下：

华章网站：www.hzbook.com

电子邮件：hzjsj@hzbook.com

联系电话：(010) 88379604

联系地址：北京市西城区百万庄南街1号

邮政编码：100037



译者序

数据挖掘是一个新兴的多学科交叉领域,并且近十余年一直保持良好的发展势头。当数据的收集、组织、存储和访问等基本问题解决之后,存储在数据库中(更广泛地说,存储在计算机系统中)的数据迅速增长。人们自然希望能够分析、理解存储在计算机系统上的海量数据,为决策提供支持。数据挖掘就是在这样的背景下产生的。

数据挖掘源于数据库学科,最初被称为数据库中知识发现(KDD)。后来,数据挖掘受到统计学、机器学习、模式识别、可视化计算等学科的普遍关注,并且广泛用于科学、工程、商业、产业、医学等诸多领域。数据挖掘的发展也得益于从应用中获得动力,从众多学科的长期工作中汲取营养。

目前,市场上已经有许多数据挖掘的教材和译著。国内许多高校都在研究生层次开设了数据挖掘方面的课程,有些学校甚至为高年级的本科生开设了数据挖掘方面的选修课。然而,对于本科生而言,目前已有的数据挖掘专著和教材都比较深,不太合适。相比之下, Soman、Diwakar 和 Ajay 的这本书更适合作为本科生的数据挖掘课程教材。该书有如下特点:

- 涵盖了数据挖掘的主要内容,包括数据的预处理、分类、预测、聚类、关联和数据可视化等内容,以及用于这些数据挖掘问题的典型算法。
- 许多算法都通过例子解释,并辅以大量图示,有利于初学者理解。
- 详细介绍如何使用开源软件包 Weka 进行数据挖掘,并简略介绍微软 Excel 上的数据挖掘工具 ExcelMiner 的用法。在课堂教学的同时,配合使用这些数据挖掘软件进行实验有助于提高学生的学习兴趣,加深对数据挖掘理论和算法的理解。
- 介绍一些数据集。这些数据集取自 UCI 机器学习库,已经成为研究算法性能的基准数据集,可以用作实验数据集。

全书共有 12 章。范明翻译了第 1~8 章,牛常勇翻译了第 9~12 章和附录。在翻译的过程中,我们对原书中明显的笔误和印刷错误进行了更正。

译文中的错误和不当之处,敬请读者朋友指正。意见和建议请发往 mfan@zzu.edu.cn, 我们不胜感激。

译者

2008 年 8 月于郑州大学



www.hzbook.com
hzbook.com
1010-82729004
100037

前言

在大规模数据集快速增长的今天,数据挖掘应成为一个新的学科。我们生活在这样一个世界,即便是简单的日常任务,如打电话、使用信用卡或购买五金电器和杂货,都会留下电子印记。诸如核物理和天体物理领域科学实验数量的增加导致每月可能产生几 PB (petabytes) 规模的数据。近来,生命科学正在成为数据驱动的科学。

广泛用于商业和上述科学领域中的自动数据收集设备每小时能够产生几 TB (terabytes) 规模的数据,致使已有的推理方法过时。世界上最大的数据仓库——Walmart 系统包含 500 TB 数据。它实在太大了,以至于无法装入任何计算机的内存。数据挖掘技术的产生源自进行数据处理业务的企业和进行数据处理研究的科学家需要找到有效的模式来自动处理海量数据。模式可以是简单的数据汇总、数据划分或数据内部的依赖模型。作为知识发现过程,数据挖掘旨在从原始数据得到“被证实的知识”。

认识到这种新技术对科学和商业的重要性之后,世界上大部分大学都开设了通用的数据挖掘原理课程和针对生物信息学、商务智能、卫生保健管理等领域的数据挖掘课程。

本书旨在为读者介绍数据挖掘方法和算法,使读者可以使用这些方法解决现实世界中的问题。书中包含了数据挖掘领域广泛使用的大部分方法(即印度和美国顶级大学教学大纲中所列出的方法),并附以简单的例子,因而是理想的课堂学习教材。像神经网络和归纳逻辑程序设计这样的主题,本书没有包括在内,因为已经有很多好的神经网络方面的教材;而对于归纳逻辑程序设计,则需要很好的逻辑程序设计方面的预备知识。作为省略神经网络内容的补偿,我们增加了一章来介绍支持向量机(SVM)。SVM 在数据挖掘中的使用日益广泛。在许多情况下,SVM 的分类性能比经典的神经网络好。

数据挖掘方面的大多数教材都很关注理论,对实际例子或实际数据的关注度不够。为了弥补这一缺点,本书包含了一些解答例子和在一些实际数据集上进行数据挖掘的结果。随书光盘中提供了大量数据集,还介绍了如何使用两种最广泛应用的软件:Waikato 大学的 Weka 和美国 Quantlink 公司的 ExcelMiner。Weka 作为开源代码提供,而 ExcelMiner 可以下载、免费使用一个月。教材和随书光盘中的介绍为采用本书的教师提供了足够的指导,以便教师在实验室进行数据挖掘实践。

为了使得理论讲解更加生动,书中包含许多图,并且在一些案例中给出一系列图,用于解释随算法执行参数如何变化。之所以采用这种方法是考虑到与工程领域的其他学科的学生相比,计算机专业的学生在高等数学和统计学方面的实践较少。我们希望读者通过本书可以比较轻松地理解数据挖掘。

本书共有 12 章。第 1 章从数据挖掘的基本介绍开始,使用了一些取自不同领域的成功例子。该章旨在概述这种新技术,并吸引学生进行实际数据挖掘项目。该章还给出数据挖掘过程的大致步骤,并进一步解释数据挖掘的主要挑战。

第 2 章从商务角度讨论数据挖掘,在此通常假定数据是商业事务的结果。这一章从数据挖掘的历史和演化开始,进而讨论数据仓库、联机分析处理(OLAP)和决策支持系统(DSS)的异同,以及它们最终发展到今天的数据挖掘。

第 3 章介绍表示对象的数据的类型和格式。通常,我们用一些对象上的观测/测量的特征来

抽象对象,而对象集就成为数据数组(表)。特征的选取和所研究的问题有关。大部分数据挖掘算法采用这种表格形式。该章还进一步解释诸如决策树、聚类和关联规则挖掘等主要数据挖掘算法的输出形式。

第4章介绍广泛使用的数据挖掘工具之一——决策树构造。该章将解释诸如ID3(C4.5)、CART、CHAID这样的算法。然后,将继续解释树剪枝的必要性和方法,以及各种模型评估技术。该章最后介绍关于代价敏感学习的注记。

进行数据挖掘的人会将90%的时间用于数据预处理,只将约10%的时间用于数据挖掘方案和输出评估。第5章将详细讨论预处理的必要性和预处理的主要步骤。

在过去的几年中,我们已经开发了许多数据挖掘算法。为了按照一定标准评估这些算法,可以从因特网下载一些标准数据集。第6章给出其中一些标准数据集的描述和某些数据挖掘工具在这些数据集上的输出。

关联规则挖掘是市场营销研究领域广泛引用的数据挖掘工具。第7章将用简单的例子介绍Agrawal和Srikant的著名算法——Apriori算法。

没有实际实验的数据挖掘可能相当枯燥,因此,第8章将专门介绍使用像Weka这样的开源软件包进行实际数据挖掘的方法。美国的大部分大学都使用这种工具进行课堂教学。最近,另一种工具开始在管理专业的学生中流行,这就是ExcelMiner——一种添加到微软Excel上的数据挖掘工具。我们也将讨论如何使用这种工具进行数据挖掘。本书所附的光盘中包含上述软件的使用说明。

第9章将介绍一些经典的统计学技术,如用于分类的朴素贝叶斯、最近邻回归方法。该章还包含一些现代工具,如遗传算法和数据挖掘应用的GMDH方法。

支持向量机(SVM)是数据挖掘领域最热门的课题之一,我们专门用一章的篇幅来讨论这部分内容。在第10章,我们从SVM的线性规划(LP)公式开始,简要介绍SVM。由于LP求解程序包含在微软的Excel中,因此可以使用它求解SVM。然后,我们考虑SVM的一种变形,称作近支持向量机(Proximal Support Vector Machine, PSVM),它也能用Excel求解。PSVM的优点是这种非线性版本非常容易求解,用6行Matlab代码就足以求解PSVM。

第11章将介绍另一种主要的数据挖掘工具——聚类技术。我们试图给出聚类概念和算法的非常简单和清晰的描述。所涉及的算法包括层次聚类、k-均值、k-中心点、DBSCAN、OPTICS、BIRCH、COBWEB、CHAMELEON和基于图的技术。

第12章是数据可视化的基础。多维数据可视化本身是一个专门领域。本章只介绍一些基本的方法。

我们热切地希望本书能够使读者对数据挖掘这个令人激动的、迅速发展的领域产生兴趣。我们希望得到读者的指正、劝告、建议和建设性批评。

K. P. Soman

Shyam Diwakar

V. Ajay

目 录

出版者的话

译者序

前 言

第1章 数据挖掘 1

1.1 引言 1

1.1.1 数据挖掘与知识发现 1

1.1.2 数据挖掘与数据分析 2

1.1.3 数据挖掘与统计学 2

1.1.4 数据挖掘与机器学习 3

1.2 数据挖掘——成功的例子 3

1.3 数据挖掘研究发展的主要原因 9

1.4 当前研究成果 9

1.5 图形模型和层次概率表示 10

1.6 新的应用 10

1.7 影响数据挖掘的趋势 11

1.8 研究挑战 12

1.9 实验平台和基础设施 13

参考文献 13

第2章 从商务角度看数据挖掘 15

2.1 引言 15

2.2 从数据挖掘工具到解决方案 16

2.3 数据挖掘系统的演变 17

2.4 知识发现过程 18

2.5 数据挖掘支撑技术概述 18

2.5.1 数据挖掘：验证与发现 19

2.5.2 决策支持系统 19

2.5.3 OLAP 20

2.5.4 桌面 DSS 21

2.5.5 数据仓库 21

2.5.6 数据挖掘过程 22

2.6 数据挖掘技术 24

参考文献 25

第3章 数据挖掘算法的数据类型、

输入和输出 26

3.1 引言 26

3.2 实例和特征 26

3.3 特征(数据)的不同类型 27

3.4 概念学习与概念描述 28

3.5 数据挖掘的输出——知识表示 30

3.5.1 分类学习算法的知识输出 30

3.5.2 聚类学习算法的输出 33

3.5.3 关联规则的输出 36

3.5.4 用于数值预测的树的输出 36

3.5.5 基于实例的学习和知识表示 38

参考文献 39

第4章 决策树——分类和回归树 40

4.1 引言 40

4.2 构造分类树 42

4.2.1 用于标称属性的 ID3 算法 42

4.2.2 信息论和信息熵 43

4.2.3 构造树 44

4.2.4 高分支属性 48

4.2.5 从 ID3 到 C4.5 49

4.2.6 形象化地理解 ID3 和 C4.5

算法 49

4.3 CHAID 51

4.3.1 CHAID 的数学工具 52

4.3.2 CHAID 变量的类型 52

4.3.3 CHAID 算法 52

4.3.4 CHAID 算法描述 53

4.3.5 将 CHAID 用于气象数据 54

4.3.6 单调变量的预测子级别合并 56

4.4 CART(分类和回归树) 57

4.4.1 CART 使用的不纯度度量	57	5.5.3 缺失数据的机制	95
4.4.2 Gini 指数	57	5.5.4 缺失数据的机制——一个人工 例子	95
4.4.3 使用 Gini 指数——一个例子	58	5.6 在决策树归纳中处理缺失数据的 例子	97
4.4.4 双化指数	59	5.7 后处理	99
4.4.5 有序双化	60	参考文献	100
4.4.6 CART 分析的步骤	60		
4.5 回归树	60	第 6 章 数据集	102
4.5.1 回归树的一个例子	60	6.1 引言	102
4.5.2 基于树的回归	61	6.2 隐形眼镜	102
4.5.3 最小二乘方回归树	63	6.3 鸢尾属植物数据库	104
4.5.4 LS 回归树的有效生长	65	6.4 乳腺癌数据库	106
4.5.5 连续变量上的划分	66	6.5 工资数据库	109
4.5.6 离散变量上的划分	67	6.6 信用卡数据库	110
4.5.7 模型树	68	6.7 住宅数据库	111
4.6 具有未知类值数据的类预测的 一般问题	69	6.8 1985 年汽车进口数据库	114
4.7 剪枝导论	71	6.9 徽章问题	117
4.8 模型评估	77	6.9.1 问题描述	117
4.8.1 交叉确认: 保持方法	78	6.9.2 部分数据	118
4.8.2 模型比较	79		
4.8.3 代价敏感的学习	80	第 7 章 关联规则挖掘	120
习题	80	7.1 引言	120
参考文献	84	7.2 事务数据库中关联规则的自动 发现	120
第 5 章 数据挖掘的预处理和 后处理	85	7.3 Apriori 算法	123
5.1 引言	85	7.4 缺点	127
5.2 数据预处理的步骤	85	习题	127
5.3 离散化	86	参考文献	129
5.3.1 人工方法	87		
5.3.2 分箱	87	第 8 章 用开源和商业软件进行机器 学习	130
5.3.3 基于熵的离散化	88	8.1 用 Weka 进行机器学习	130
5.3.4 找出分割点的其他简单方法	89	8.1.1 开始	130
5.4 特征提取、选择和构造	91	8.1.2 装入数据	132
5.4.1 特征提取	92	8.1.3 选择或过滤属性	134
5.4.2 特征选择	94	8.1.4 离散化	135
5.4.3 特征构造	94	8.1.5 关联规则挖掘	140
5.5 缺失数据及其处理方法和技巧	94	8.1.6 分类	142
5.5.1 什么是缺失数据	95	8.1.7 聚类	146
5.5.2 缺失数据的主要原因	95	8.2 XLMINER	150

参考文献	150	10.5 生成数据集	202
第9章 分类和回归算法	151	10.5.1 螺旋数据生成器	202
9.1 引言	151	10.5.2 棋盘格数据集	203
9.2 朴素贝叶斯	151	10.5.3 多元正态分布数据生成器	204
9.2.1 朴素贝叶斯的零频率问题	153	10.6 问题及解答	206
9.2.2 缺失值和数值属性	153	习题	207
9.3 多元回归分析	155	参考文献	207
9.3.1 什么是回归分析	155	第11章 聚类分析	209
9.3.2 简单和多元回归分析	155	11.1 引言	209
9.3.3 在市场营销中的应用	155	11.1.1 相似性及其度量	211
9.3.4 方法	155	11.1.2 聚类的基本类型	218
9.3.5 使用 Excel 进行多元回归分析	156	11.2 划分聚类	230
9.3.6 输入数据	156	11.3 k-中心点	233
9.3.7 回归输出	158	11.4 现代聚类方法	234
9.4 逻辑斯谛回归	160	11.5 BIRCH	236
9.5 k-最近邻分类	163	11.6 DBSCAN	238
9.5.1 k-近邻预测	165	11.6.1 DBSCAN 算法的概念	239
9.5.2 k-NN 算法的缺点	165	11.6.2 DBSCAN 的基本概念和算法	240
9.6 GMDH	166	11.6.3 算法	241
9.6.1 引言	166	11.6.4 DBSCAN 算法的优点	242
9.6.2 数据处理群组方法的背景	166	11.7 OPTICS	242
9.6.3 构建决策规则	168	11.7.1 引言	242
9.6.4 实验结果	171	11.7.2 OPTICS 算法的动机	243
9.6.5 讨论和总结	171	11.7.3 OPTICS 采用的概念	243
9.7 进化计算和遗传算法	171	11.7.4 OPTICS 算法	243
9.7.1 进化理论	172	11.7.5 可达图	250
9.7.2 遗传算法	175	11.7.6 优点	252
9.7.3 使用遗传算法进行机器学习	177	11.7.7 缺点	252
习题	178	11.8 基于图划分的聚类	252
参考文献	180	11.8.1 加权图划分	252
第10章 支持向量机	182	11.8.2 平衡图划分——基本原则	253
10.1 引言	182	11.8.3 k 路划分	256
10.2 线性支持向量机的基本思想	185	11.9 CHAMELEON: 两阶段聚类	
10.3 软边缘 SVM: 线性核	187	算法	256
10.3.1 线性 SVM 的线性规划公式表示	189	11.9.1 数据建模	257
10.3.2 有训练误差的 SVM: 非线性核	190	11.9.2 簇相似性建模	257
10.4 邻近支持向量机	190	11.9.3 CHAMELEON 的两个阶段	258
		11.9.4 用例子说明 CHAMELEON 算法	259

11.10 COBWEB 概念聚类算法	262	12.2 多维可视化的图表表示	294
11.10.1 COBWEB 算法	262	12.2.1 kiviati 图	294
11.10.2 COBWEB: 一个简单 例子	264	12.2.2 平行坐标系	295
11.11 GCLUTO: 图形化聚类工具箱 ...	270	12.2.3 3D 散点图	295
11.11.1 概述	271	12.2.4 3D 曲线图	296
11.11.2 GCLUTO 中的可用选项	277	12.2.5 体积透视图	296
11.11.3 使用 GCLUTO 进行文本 挖掘	283	12.2.6 房图	297
习题	285	12.2.7 Chernoff 脸图	298
参考文献	291	12.3 可视化数据挖掘	298
第 12 章 多维数据可视化	292	参考文献	299
12.1 引言	292	附录 A SVM 公式: 完全可分的线性 分类器	300
		附录 B 图划分的矩阵形式	304

第1章 数据挖掘

1.1 引言

计算机科学家经常提到摩尔定律：计算机的处理速度大约每 18 个月翻一番。但是很少有人知道计算机的存储容量大约每 9 个月翻一番 [1] (Goebel 和 Gruenwald 1999)。像理想气体一样，计算机的数据库迅速膨胀，占满了可用的存储空间，导致数据库中的大量数据成为未开发利用的资源。这些数据就像一个金矿，可以从中提取信息。然后，利用数据挖掘技术，可以将这些信息转换成有价值的知识。

很难说清楚有多少存储在全世界公司、学校、政府部门和其他机构的大型数据库中未使用的海量数据以及其当前增长率。据估计，美国国会图书馆存储的信息量高达 3PB [2] (Lesk 1997)。Lesk [2] 估计，全世界每年大约产生 160TB 信息。而且，他估计已售出的磁盘空间将超过十万 TB。很快，计算机的数据存储容量将超过人们使用该数据存储和使用其中数据的能力。将海量数据转换为知识的过程将变得价值无限。为此，在过去的 10~15 年中，一种称作数据库中知识发现 (KDD) 的过程逐步发展完善。数据挖掘算法就包含在 KDD 过程中。

典型的数据库用户使用一种界面通过诸如 SQL 这样的标准技术从数据库中检索数据。数据挖掘系统将这一过程向前推进一步，支持用户从数据中发现新的知识 [3] (Adriaans 和 Zantinge 1996)。按照计算机科学家的观点，数据挖掘是一个多学科交叉领域。诸如神经网络、遗传算法、回归、统计分析、机器学习和聚类分析等数据处理技术经常出现在数据挖掘文献中。许多研究者认为数据挖掘还不是一个完善的学科，数据可扩展性、与数据库系统的兼容性，以及可用性和准确性都有待改进。

1.1.1 数据挖掘与知识发现

对于数据挖掘和知识发现，大部分作者都有不同的定义。Goebel 和 Gruenwald [1] 将 KDD 定义为识别数据中有效的、新颖的、潜在有用的和易于理解的模式的非平凡过程，而将数据挖掘定义为观测数据中的模式或模型提取。Berzal 等 [12] 将 KDD 定义为隐含 (先前未知的) 信息的大量数据中潜在有用信息的非平凡提取。Goebel 和 Gruenwald 的 KDD 模型表明数据挖掘是如下所示的整个 KDD 过程的一个步骤：

- 1) 确定和逐步理解应用领域。
- 2) 选择所研究的数据集。
- 3) 选择补充数据集。集成这些数据集。
- 4) 数据编码、清理重复和错误数据、变换数据。
- 5) 开发模型、构建假设。
- 6) 选择合适的数据挖掘算法。
- 7) 解释结果。使用合适的可视化工具显示结果。

⊖ 表示引用章后参考文献 1。在本书中将统一用此种方式表示对章后参考文献的引用。

8) 检验结果。

9) 管理发现的知识。

尽管数据挖掘只是 KDD 过程的一部分,但是数据挖掘技术提供了推动 KDD 过程的算法。前面展示的 KDD 过程是一个永不休止的过程。数据挖掘是 KDD 过程的核心。如果讨论数据挖掘,则应当理解为讨论 KDD 过程。在本书中,我们关注数据挖掘算法。

Adriaans 和 Zantinge[3]强调 KDD 界将术语数据挖掘专指 KDD 过程的发现阶段。他们对 KDD 的定义如下:KDD 是数据中蕴涵的、先前未知的和潜在有用的知识的非平凡提取。类似地,Berzal 等[12]将数据挖掘定义为一般术语,涵盖用于从大型数据库中提取有用信息的研究成果、技术和工具。Adriaans 和 Zantinge[3]还指出,KDD 从专家系统、机器学习、统计学、可视化和数据库技术汲取营养。

1.1.2 数据挖掘与数据分析

Comaford(“计算国际公司”CEO,该公司是一家专门从事 GUI 设计和客户/服务器开发的咨询公司)指出关于数据挖掘有一些误解。按照 Comaford 的观点,数据挖掘不同于数据仓库或数据分析。数据挖掘是一个动态过程,能够比数据分析更智能地使用数据仓库。数据挖掘构建模型,该模型可以用来进行预测,而不需要附加的 SQL 查询。数据挖掘技术既适用于小型数据集也适用于超大型数据集。我们不仅要考虑数据集的大小,而且必须将适当的宽度(width)、深度(depth)和体积(volume)作为三个重要要求加以考虑。有效的数据挖掘需要数据库记录的许多属性(宽度)、大量的数据库实体的实例(深度)和数据库设计所确定的许多实体(体积)的记录。数据挖掘最适合面向顾客的应用,而不是一般的商务应用。数据挖掘不一定需要人工智能(AI)。如果数据挖掘算法使用 AI,则对用户应当是透明的。换句话说,除面向顾客的应用之外,Comaford 并没有把数据挖掘看作一般商务工具。对于商业数据挖掘应用,这种看法是对的。这种看法强调用于技术数据的数据挖掘应用需要。

关于宽度、深度和体积,Adriaans 和 Zantinge[3]的观点与 Comaford 不同。按照 Comaford 的观点,通过将数据库的感兴趣属性分散到相关记录集中,连接操作消除了对体积定义的需要。另一方面,Adriaans 和 Zantinge 将数据挖掘视为多维数据空间的探查。考虑一个具有一个实体和 100 万个记录的数据库。如果该数据库只有一个属性,则它只有一个维。假设该维的刻度从 0~100,具有 1/100 的分辨率。在一维情况下,对于 100 万个记录,每单位空间或单位长度平均 10000 个记录。对于两个属性(两个维),每单位面积平均 100 个记录。对于 3 个属性,每单位体积平均只有一个记录。为了观察这个数,考虑真空每立方英寸包含一到两个原子[4](Elert 2000)。这样,具有 100 万个记录的 3 维数据挖掘空间是一个密度非常低的空间。进一步,如果数据库具有 10 个属性,则记录的密度为每单位超体积 10^{-14} 个记录。这一类比表明,随着属性数目增加到 3 以上,即使对于很大的数据库,多维空间也变得相对很空。因此,超空间中记录的密度是选择数据挖掘技术的一个考虑因素。

1.1.3 数据挖掘与统计学

在 20 世纪上半叶,统计学家主要分析系统规划的实验,回答完全以公式化方式表达的科学问题。这些实验产生的高质量的数据很少。在这些受控的条件下,人们通常可以得到收集和分析数据的最佳方法,并可以(数学地)证明这种性质。

在 21 世纪初期,情况变得有些混乱。数据集的规模发生了变化。数据沿两个方向增长:不仅观测值越来越多,而且变量也越来越多。通常,这些数据不是直接被抽样的(为了分析),而是其他活动的副产品。这样,它们并非源于好的实验设计,并且某些变量可能并不

包含信息。因此,数据中的“噪声”越来越多。

因此,数据挖掘在许多方面都不同于传统的统计学:形式的统计学推断是假设驱动的,即形成假定并在数据上验证它。相比之下,数据挖掘是发现驱动的,即自动地从数据中提取模式和假定。换句话说,数据挖掘是数据驱动的,而统计学是人驱动的。与数据挖掘类似的统计学分支是探测式数据分析,尽管像统计学其他领域一样,该领域关注的数据集比数据挖掘研究者关注的数据集小得多。

数据挖掘的目标也不同于传统的统计学。有时,其目标是提取可以容易转换成逻辑规则或可视化表示的定性模型。从这个意义上说,数据挖掘是以人为中心的,并且时常与人机界面研究结合在一起。

1.1.4 数据挖掘与机器学习

机器学习研究构建由经验学习的系统。机器学习算法旨在处理一些问题领域,这些问题领域中没有好的理论模型,但可以进行经验观测。例如:

- 1) 零售商想知道把哪个顾客群或个人作为广告宣传对象。
- 2) Madhuri Dixit 的狂热追随者想在大型图像数据库中找出她的所有照片。
- 3) 科学家想知道导致癌症在家族中蔓延的基因。

机器学习算法是用于处理这类问题的计算机应用的核心,因此它为数据挖掘提供了技术基础(模式发现引擎)。

此外,机器学习范型(MLP)“让数据决定模型”是统计学范型“让数据拟合诸如逻辑斯缔回归这样的预先确定的方程”的一种实际替代。当数据很“小”时,统计学范型仍是数据挖掘的基础。将小数据拟合为严格参数化的、假定的模型,过去是并且现在仍然是合理的选择。然而,电脑空间需要一种范型转换来处理大数据。

1.2 数据挖掘——成功的例子

1) 贝尔大西洋公司(Bell Atlantic)[6]:当顾客向贝尔大西洋公司报告电话问题时,该公司必须决定派什么样的技术人员去解决该问题。从1991年开始,该公司使用专家系统做此决定。1999年,该专家系统被数据挖掘创建的一组规则取代。这些学习得到的规则每年为贝尔大西洋公司节省了1000多万美元,因为这些规则减少了他们做出的错误决定。此外,专家系统也已经进入难以有效维护阶段。由于学习得到的系统是通过在实例上训练而得到的,因此容易维护,并且容易调整以适应不同的地区和开销的变化。

2) 美国万国宝通银行(American Express)[7]:20世纪80年代,美国万国宝通银行(UK)使用统计学方法将贷款申请分成3类:肯定接受的申请、肯定拒绝的申请和需要专家判定的申请。专家预测申请者是否会拖欠贷款的准确率只能达到50%。机器学习产生的规则对这种情况预测的准确率可达到70%,并能立即投入使用。

3) 英国石油公司(British Petroleum Corporation)[7]:从地下抽取出的原油通常混有天然气,并且必须在炼油前将二者分离。找到控制分离过程的理想参数是一项复杂的任务。英国石油公司使用机器学习技术创建了一组设定控制参数的规则,使得专家需要一天多才能完成的任务在10分钟内就能完成。

4) R. R. Donnelly(一家美国大型印刷公司)[8]:在凹版印刷时,印刷滚筒上有时会出现凹槽,毁坏最终产品。当出现这种情况时,必须停止生产,修理或更换滚筒,然后再重新开始印刷。即便对于专家,条带产生的原因也尚未完全清楚。R. R. Donnelly印刷公司雇了一个顾问,就减少条带问题听取其建议,同时使用机器学习为控制过程参数(例如,油墨

黏性)创建规则,减少条带。学习得到的规则(见图1-1)在以下几方面都优于顾问的建议:

- ①学习得到的规则更适合收集训练数据的工厂。
- ②学习得到的规则更完整,填补了顾问建议的空白。
- ③一条与顾问建议相抵触的学习得到的规则被证明是正确的。

学习得到的规则已经在田纳西的加勒廷工厂使用了十多年,并将条带出现的次数从538次(1989年)降低到26次(1998年)。

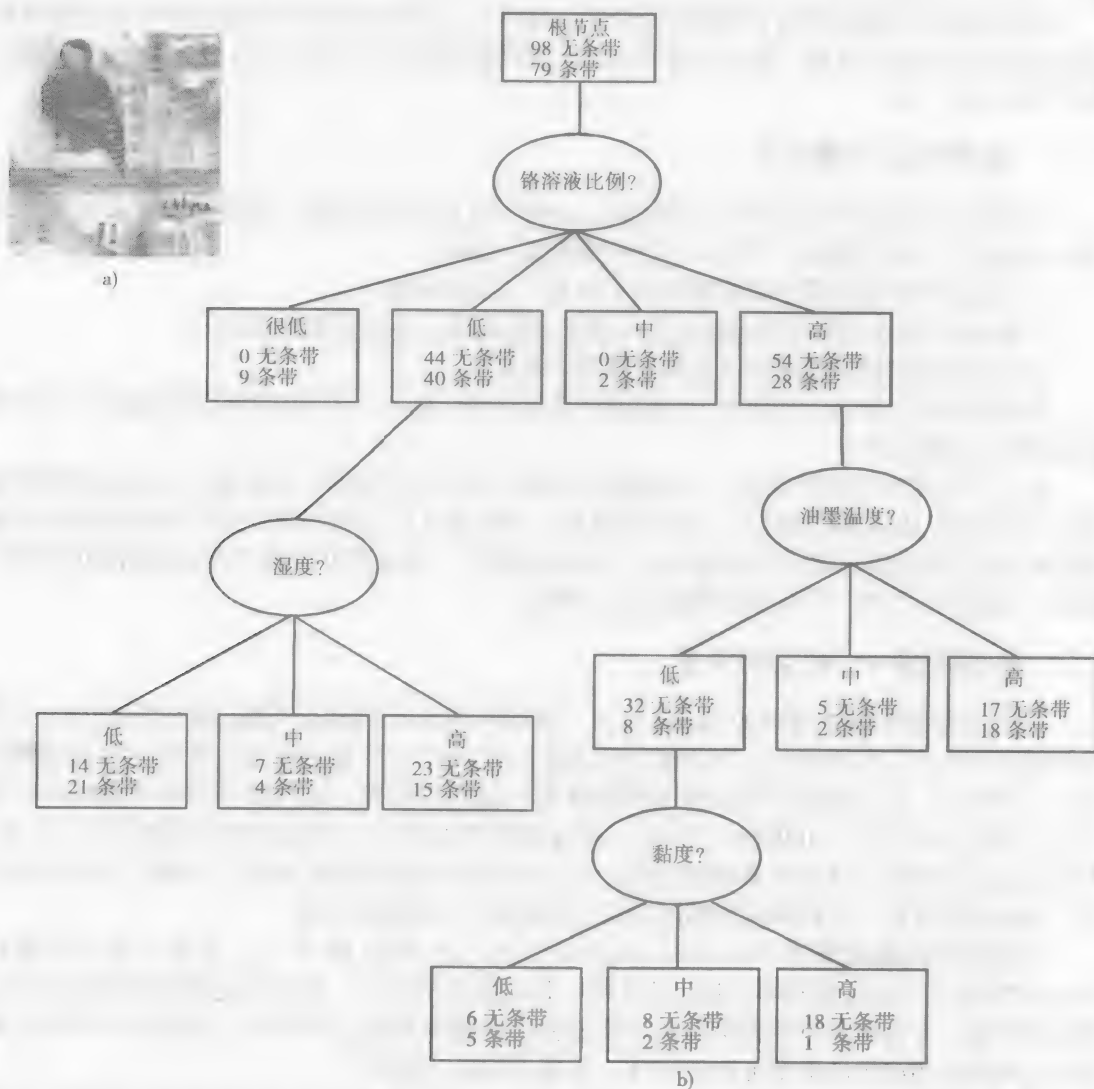


图1-1 a)在称作条带的著名滚筒印刷问题中,油墨条纹毁坏印刷页面。b)决策树。一棵帮助 R. R. Donnelly 和 Sons 公司的印刷操作工确定何时可能出现条带的决策树。为了创建该决策树,计算机算法考察了500次印刷数据。结果发现,预测条带出现的最佳属性是电镀液中的铬比例。在后面的节点,算法发现了帮助操作工评估条带出现风险的其他属性,并相应地添加一些分支到树上。在某些称作叶节点的节点上,不能再划分分支,因为所有的数据都属于一个类,或者因为没有其他可用的信息来准确预测出是否有条带。像其他数据挖掘方法一样,如果没有模式的话,决策树不能发现模式。这里显示的是包括177次印刷数据的决策树的子树。该树表明,一个使条带最小化的解可能是高铬溶液比例、低油墨温度和低油墨黏度。

5) 飞行模拟和学习: 通过数据挖掘学习控制规则提供了一种快速、简便地构建复杂的控制系统的新方法。Claude Sammut 和 Donald Michie[5]使用飞行模拟程序记录专家驾驶飞机的动作。然后, 将日志文件作为一个称作决策树归纳(decision tree induction)的数据挖掘算法的输入。归纳程序的输出质量则通过运行自动驾驶方式的模拟程序检验, 其中自动驾驶程序代码由归纳程序产生的决策树导出。图 1-2 显示了归纳程序产生的一些规则。

```

airspeed > 127 : thrust_100
airspeed <= 127 :
| X_feet > 121.33 : thrust_30
| X_feet <= 121.33 :
| | elevation <= -43 :
| | | Z_feet > -11514.8 : thrust_0
| | | Z_feet <= -11514.8 :
| | | | climb_speed <= -13 : thrust_0
| | | | climb_speed > -13 :
| | | | | Z_feet > -18475.8 : thrust_10
| | | | | Z_feet <= -18475.8 :
| | | | | Y_feet <= 1535.21 : thrust_20
| | | | | Y_feet > 1535.21 : thrust_10
| | | elevation > -43 :
| | | Y_feet <= 638.76 : thrust_25
| | | Y_feet > 638.76 :
| | | | Z_feet <= -26230.1 : thrust_15
| | | | Z_feet > -26230.1 : thrust_20

```

图 1-2 从飞行模拟和学习软件学习的规则

现在, 科学家正在使用这种方法理解难以自省的亚认知技能。例如, 如果问你如何骑自行车, 你可能给出令人满意的回答, 因为这种技能是下意识地学习和实施的。通过监控下意识的技能的性能, 我们可以用符号规则的形式构造该技能的功能描述。这不仅揭示了该技能的本质, 而且也可以用作训练的辅助支持, 因为可以清楚地向学员展示他/她正在做什么。

6) 机器人学习: 用类似的风格, Claire D'Este、Mark O'Sullivan 和 Nicholas Hannah[9]展示了机器人如何快速学习目标追踪和障碍回避: 操作员通过操纵杆控制机器人的运动。为了监督学习, 操纵杆命令被离散化为有限个类(如右易、左难), 用来控制实际的机器人为操作员提供反馈, 并记录此时由机器人接收到的传感器信息(见图 1-3)。例如, 机器人观察到目标在左边, 并且操纵杆命令是“左易”。当操作员操纵机器人时, “离散命令”的使用确保使用操纵杆可以精确控制而不损害知识库。由这些数据创建的文件反馈给学习算法, 以传感器信息为输入特征, 实际命令为期望的输出分类。符号学习算法(C4.5 决策树, 更多细节见第 4 章)和连接者学习算法(后向传播神经网络)用来创建知识库, 再用知识库来控制机器人, 而不需要人的干预。图 1-3a 显示了专家支持下的规则产生, 而图 1-3b 显示了如何使用知识库控制机器人。

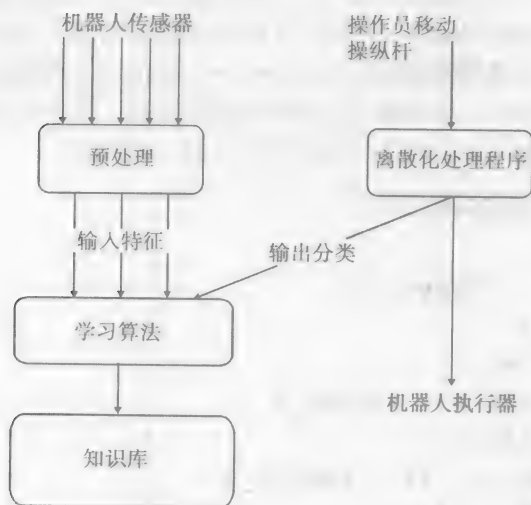


图 1-3a) 进行机器人学习的规则产生

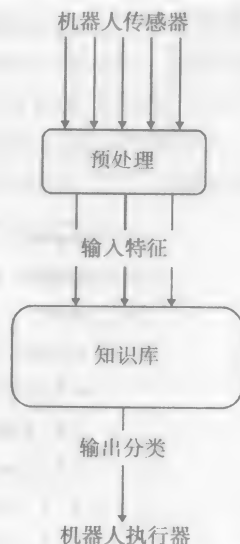


图 1-3b) 用知识库控制机器人

7) 计算机控制道路车辆：NavLab (www.navlab.org) 计算机控制的车辆 (卡内基梅隆大学, CMU) 通过观察驾驶员学习独立地在公路上驾驶。CMU 的 Navlab 小组建造了许多计算机控制的车辆, 用来研究自动和辅助驾驶。自 1984 年以来, 他们建造了一系列机器人小汽车、有篷货车、SUV 和公共汽车。Navlab 家族的最新产品是 Navlab 11。这是一种 Wrangler 机器人吉普, 装备有用于短程和中程障碍检测的各种传感器。图 1-4 显示了 Navlab 11。



图 1-4 Navlab 计算机控制的车辆

8) 学习赢得十五子棋游戏：TD-Gammon (<http://www.research.ibm.com/massive/tdl.html>) 的十五子棋游戏程序 (Gerald Tesauro/IBM) 使用增强学习, 通过让玩家在大师级难度级别与自己对弈 100 多万次, 来学习玩十五子棋游戏。这是世界顶级的十五子棋游戏程序。

9) 学习过程控制：制造核燃料球的威斯丁豪斯过程被大量以复杂方式相互影响的控制参数控制。不正确的设置, 产量和回报都很低。使用机器学习创建了一组控制生产过程的规则, 1984 年使用以来, 每年为威斯丁豪斯增加 1000 多万美元的生意额。

10) 试管婴儿：试管婴儿涉及从妇女卵巢中收集多个卵子, 与伴侣或捐赠者的精子授精后产生多个胚胎。从中选择某些胚胎, 植入妇女的子宫中。在这个过程中, 问题是如何选择“最好的”胚胎——最可能存活的那些胚胎。选择是基于胚胎的大约 60 个特征做出的, 包括

刻画它们的形态、卵母细胞、卵泡和精子样本。特征的数目非常多,胚胎学家很难同时评估它们,并将历史数据与胚胎是否导致活的婴儿的关键结果建立联系。在英格兰的一个研究项目中,用胚胎和它们的结果的历史记录作为训练数据,使用机器学习研究选择技术。

11) **养牛**: 每年,新西兰奶牛场主必须做出棘手的决定: 哪些奶牛留在奶牛场, 哪些奶牛卖到屠宰场。通常, 随着饲料储备减少, 在产奶季节将近结束时, 奶牛场大约淘汰 1/5 的奶牛。每头奶牛的血统和产奶史将影响这一决定。其他因素包括年龄(奶牛的产奶期在第 8 年将近结束)、健康问题、难产史、不期望的暴躁脾气(踢、跳栏)、不再怀小牛等。对于数百万头奶牛, 每头奶牛大约记录 700 个属性值。用机器学习来研究成功的农场主在做决定时考虑哪些因素——不是为了自动决策, 而是为了将他们的技能和经验传授给别人。机器学习是用于从数据中挖掘知识的新兴技术, 也是一项许多人都开始认真对待的技术。

12) **分子生物学**: 近年来, 我们见证了基因序列信息(包括整个基因组序列)的加速累积。这种数据分析的第一步是识别每个新的基因组中数以千计的基因。Glimmer 是一个基于机器学习的程序, 它能找出基因组中 97% ~ 99% 的基因, 而不需要人的干预。

13) **药物发现**[10]: 为了设计具有期望生物活性的新药, 或为了理解已知药物的活性(或非活性, 如无毒性的)的机理, 需要找到化学结构与感兴趣的活性之间的联系。从试验数据中发现的联系称作 **SAR**(Structure Activity Relationship, 结构活性联系)。由于具有复杂的 3-D 形状, 除非最简单的情况, 否则人工的 SAR 分析是不可行的。归纳逻辑规划(ILP)是一种数据挖掘技术, 已证明它对于发现 SAR 特别有用, 因为除了药物的物理化学性质之外, 它还能直接推断药物的 2-D 或 3-D 结构。ILP 重新发现已知的 SAR 和提出新的 SAR 的专门应用有苏拉明衍生物、治疗早年痴呆(Alzheimers)疾病的药物、二氢叶酸还原酶和磷酸化酶抑制药物以及诱变性和致癌性的毒物学模型。

14) **制药**[10]: 科学家和制药公司都感兴趣的一项重要任务是发现某类蛋白质的新成员, 如人类神经肽原(NPP)。为了确定一种给定的化合物是否是 NPP, 必须合成它并经过生物学测试。如果可以通过自动方法对可能的 NPP 预先进行精确地过滤, 则可以节省大量的时间和金钱。由于 NPP 的多样性, 基于序列相似性的标准方法得到的结果并不是特别准确。基于 ILP 方法非常奏效。此外, 不像标准的过滤方法, ILP 产生的规则将特定的生物学机能与 NPP 族相关联。这些看上去相当合理, 现在正被 SmithKline Beecham 的科学家研究。

15) **天文学**[11]: 帕洛马第二星空观测天文台(POSS-II)产生了 3000 幅图像, 包含数十万星系、恒星、类星体和科学家感兴趣的其他天体。先前观测的图像中的天体通过肉眼进行识别和分类。对于 POSS-II, 这种方法是不可行的, 因为数据的绝对数量太大, 而且要识别和分类的对象太暗, 难以通过肉眼看到。使用机器学习开发了一个完全自动的分类系统, 即使对于所考察的最暗的天体, 准确率也超过 92%。准确率水平完全能被使用它的科学家所接受。

16) **医学**[11]: 通过使用学习技术, 分析了旧金山 1991 ~ 1999 年肺结核病人的数据, 产生一个概率关系模型(PRM)。学习得到的模型的结构揭示了变量之间的直接和间接依赖。除了证实先前人工分析已知的依赖之外, 学习得到的模型还揭示了新的联系, 值得进一步进行流行病学调查。

17) **地球物理学**: 一篇出现在 2003 年 7 月的美洲日报(*American Dailies*)上新闻报道这样说: “NASA 的数据挖掘揭示了自然灾害的新历程。NASA 使用卫星数据, 详细绘制了全球过去 20 年自然灾害、人类活动和地球大气中二氧化碳含量上升的相互影响。新的结果通过一种称作‘数据挖掘’的技术得到。数据挖掘对大量卫星和科学数据排序, 以检测其他方法可能忽视的模式和事件。明尼苏达大学、加利福尼亚州立大学和 NASA 艾姆斯的旨在开发数据挖掘技术,

以帮助地球科学家发现全球碳循环和气候系统变化的联合项目负责人 Kumar 补充说。”

18) 欺诈检测: 另一篇报道说, 美国政府开始了一项大规模的数据挖掘研究计划, 称作知晓全部信息 (Total Information Awareness, TIA), 用于梳理人们日常生活中产生的大量信息 (购物记录、E-mail、电话记录、旅行安排), 寻找恐怖活动的警告模式。一位数据挖掘专家指出, 该项目的目标 (在多个数据库之间发现可疑模式, 同时尽可能减少假报警, 并保证个人隐私的安全) 的规模与“将人送到月球”类似。

19) 入侵检测: 1994 年 6 月, 俄罗斯圣·彼得堡的计算机专家 Vladimir Leonidovich Levin 潜入 CitiBank 的电子资金转账网络。在其后的 5 个月中, 他将 1000 万美元转移到加利福尼亚、以色列、芬兰、德国、荷兰和瑞士的账户中。他最终被捕, 并且大部分资金被追回, 但是这次事件暴露了我们的现代信息基础设施的弱点。随着印度和国际商务越来越依赖计算机和网络, 计算机犯罪的威胁也日益增加。检测和预防这种入侵也越来越困难。系统管理员和安全官员必须监控大型网络, 通常包括数千台计算机和数兆字节存储空间, 其中一台工作站上的一次安全违例就可能是损失数百万美元的事件。根据计算机突发事件反应小组 (计算机安全专业人员组织) 的估计, 只有 5% 受害站点察觉曾被侵入。尽管每台计算机的审计数据记录了检测入侵所必需的原始信息, 但是每天的审计信息太多 (其中大部分记录了平凡、无害的活动), 人们无法检查。

通过检测计算机使用日志中的异常模式, 数据挖掘系统可以为系统管理员标识可疑事件, 从而大大减少他们的负担。这种系统监视单用户的计算机或账户, 并逐渐得到用户典型行为的轮廓曲线。它可以根据与已知或期望模式的背离检测异常 (可能有害或滥用行为)。同时, 它还必须足够灵活, 可以接受“正常”改变导致的差异。例如, 由于用户学习使用新程序或承担新任务而导致的行为变化。图 1-5 显示的是普渡大学开发的一个这样的系统轮廓曲线输出。现在, 市场上有多种这样的基于商业入侵检测的数据挖掘系统。图 1-5 显示了智能代理如何使用用户的轮廓曲线进行入侵检测。

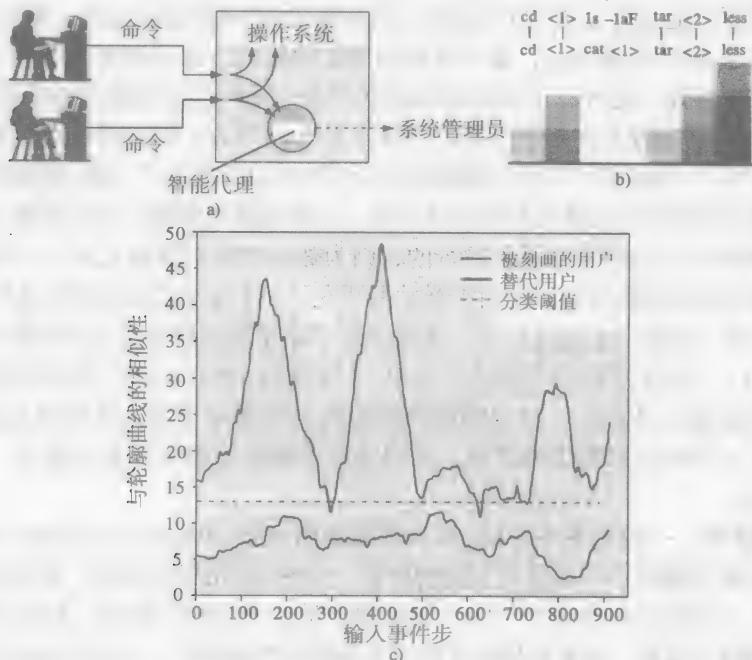


图 1-5 用于入侵检测的用户轮廓曲线

1.3 数据挖掘研究发展的主要原因

在过去的20年中,数字化的数据量正在猛增,而能够分析数据的科学家、工程师和分析人员的数量变化一直很小。为了弥合这种差距,需要全新的基础研究,这些可以分成如下主要问题:①开发挖掘大型海量和多维数据集的算法和系统。②开发挖掘新的数据类型的算法和系统。③开发挖掘分布式数据的算法、协议和设施。④提高数据挖掘系统使用的简便性。⑤为数据挖掘开发合适的隐私和安全模型。为了响应这些挑战,我们需要在数据挖掘和知识发现方面进行应用的、多学科的和交叉学科的研究。

1.4 当前研究成果

今天的数据挖掘建立在各种研究成果的基础上,其中大部分都是政府资助的研究。在本节中,我们将提到其中一些较为重要的研究。注意,其中一些研究本质上是交叉学科的研究,基于协同工作的、来自不同学科的研究者的发现。

1) **神经网络**:神经网络是一种受人脑启发而产生的系统。一个基本例子是由输入节点、输出节点和称作隐藏节点(hidden node)的中间节点组成的后向传播神经网络。最初,节点以随机权重相连接。在训练期间,使用梯度下降算法调整权重,使得输出节点正确地对提供给输入节点的数据分类。该算法独立地被多个研究小组发明。

用于分析时,训练后的神经网络可以看作一个信息分类专家。一旦经过训练,它就能对感兴趣的新情况的预测,并回答“如果……将会怎么样”问题。训练后的网络被看作一个黑箱,因为它并不提供结果的解释。其他问题包括:

①很难结合用户的干预。

②由于其迭代性质,需要超长的学习时间。

图1-6显示了一个用于癌症风险分类的神经网络的结构、输入和输出。

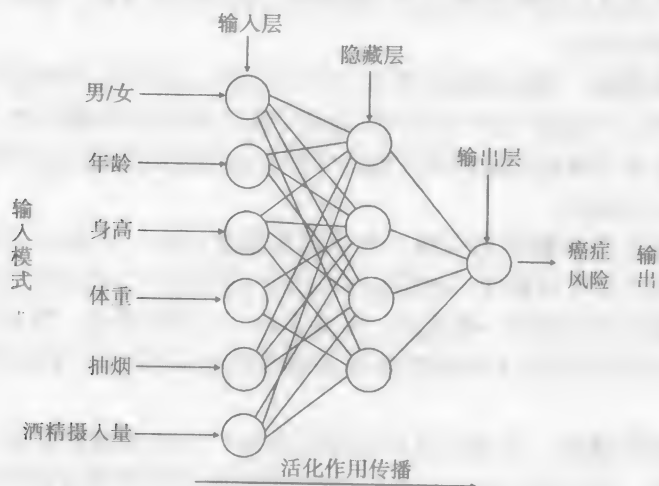


图1-6 用于癌症风险分类的神经网络

2) **支持向量机**:支持向量机是一种新算法,在诸如分类、回归和聚类等应用方面与神经网络有一争。与神经网络不同,它不是黑箱。它的分类和回归的工作可以几何地解释。支持向量机的理论的发展使得它非常适合大规模数据挖掘问题。

3) 基于树的分类方法：树是一种将大数据集分割成小数据集的便捷方式。通过将训练集置于树根，并在每个内部节点提问，通常可以非常简单地分析树叶上的数据。例如，一个预测信用卡交易是否是欺诈的分类器可以根据前一个小时是否进行了5次或更少的交易，使用一个内部节点将训练数据集划分成两个集合。经过一系列这类提问后，每个树叶可以通过简单的多数表决标记为欺诈或非欺诈。基于树的分类方法被信息论、统计学、模式识别和机器学习的研究者独立地发明。图1-7显示了一棵决策树结构。

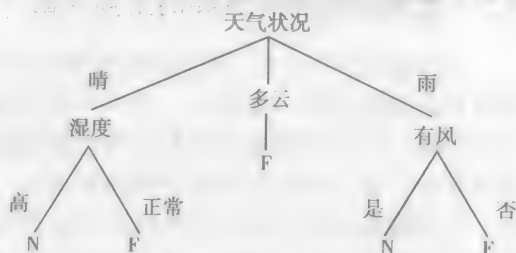


图1-7 一棵决策树

1.5 图形模型和层次概率表示

有向图是一种领域专家组织条件独立性和因果关系的定性知识的好工具。图形模型推广了马尔科夫模型和隐马尔科夫模型，被证明是一种强有力的建模工具。研究不确定性的计算概率学家和人工智能研究者独立地发明了图形模型。

1) 系综学习(ensemble learning)：通常，与其使用数据挖掘建立单个预测模型，还不如建立一组模型或模型的系综，并且使用一种简单、有效的投票策略将它们组合起来。这种简单的思想已经应用于广泛的背景和应用。在某些情况下，会认为这种技术降低了预测方差，从而降低了模型的总体误差。

2) 线性代数：可伸缩的数据挖掘算法常常高度依赖于可伸缩的线性代数基本计算。对于从文本挖掘到网络入侵检测等各种数据挖掘应用，求解线性系统的并行算法和求解高维稀疏线性系统的算法的近期工作都是非常重要的。

3) 大规模优化：某些数据挖掘算法常常可以表示为大规模的、非凸的优化问题。最近的研究提供了求解大规模连续和离散优化问题的并行和分布式方法，包括用于不能直接求解的大问题的启发式搜索方法。

4) 高性能计算和通信：数据挖掘需要在大型数据集上进行统计密集的操作。没有功能强大的 SMP 工作站和支持诸如 MPI 和 MPIO 等高性能计算协议的高性能工作站集群，这种计算是不现实的。分布式数据挖掘需要在地理分散的站点之间传递大量数据，有了广域高性能网络，这些已经可以做到。

5) 数据库、数据仓库和数字图书馆：在数据挖掘过程中，大部分时间用于为数据挖掘准备数据。如果数据已经在数据库、数据仓库和数字图书馆中，则该步骤可以简化，尽管涉及不同数据库的数据挖掘仍然是一种挑战。某些算法，如关联算法，紧紧与数据库连接在一起，而一些正在未来的数据仓库中构建的原始操作将被证明对于某些数据挖掘应用是有用的。

6) 大型数据集的可视化：通常由复杂的模拟程序产生的大型数据集，需要图形可视化方法以便更好地理解。多尺度(multi-scale)可视化的最新进展使得透视图可以快速和并行完成，使得这些可视化任务切实可行。

1.6 新的应用

数据挖掘学科在一定程度上是由新的应用驱动的。这些应用需要新的、不能被今天的技术所支持能力。这些新应用可以分成三大类。

1) 商业和电子商务数据。企业资源规划系统(back-office)、客户关系管理系统(front-office)和网络应用产生了大量关于商务过程的数据。使用这些数据进行有效的决策仍然是一项根本挑战。

2) 科学、工程和卫生保健数据。科学数据和元数据在结构上通常比商务数据更复杂。此外,科学家和工程师越来越多地使用模拟和具有领域知识的系统。

3) Web 数据。Web 上的数据不仅数量在增长,而且复杂性也在增长。现在,Web 数据不仅包括文本和图像,而且包括流数据和数值数据。

在本节中,我们将介绍每一类的一些应用。

1) 商业交易:今天,商业机构正在合并,越来越多的商业机构拥有数百万顾客和数十亿的顾客交易。他们需要知道风险(该交易是否是欺诈?该顾客是否是他的账单付费?)和机遇。(该顾客的预期利润是多少?该顾客下次最可能购买的产品是什么?)

2) 电子商务:电子商务不仅产生对于分析销售模式和风险模式的至关重要的大型数据集,而且与上面提到的一些应用不同,为了满足在线交易的需要,实时或接近实时地进行这些分析也很重要。

3) 基因组数据:基因组排序和绘制工作已经产生了大量数据库,这些数据库可以通过 Web 访问。此外,还有形形色色的其他联机数据库,包括关于疾病、细胞功能和药物信息的数据库。发现这些数据源之间的联系(大部分尚未考察)是数据挖掘的另一项根本挑战。最近,已经开发了比较整个基因组的可伸缩技术。

4) 传感器数据:卫星、浮标、气球和其他各种传感器产生了数量庞大的关于地球大气、海洋和陆地的数据。一项根本挑战是理解这些数据之间的关系,包括因果关系。例如,工业污染对全球变暖是否有影响?还有多达 1 兆兆字节到 1 千兆兆字节数据集由诸如天文学、高能物理和核物理等其他学科的传感器和设备产生。

5) 模拟数据:现在,除理论和实验外,模拟已被作为第三种科学形态。今天,不仅实验会产生大量数据集,模拟也会产生大量数据集。数据挖掘(更一般地,数据密集的计算)被证明是理论模拟和实验之间的重要环节。

6) 卫生保健:多年来,卫生保健一直是美国 GDP 增长最快的部分。医院、卫生保健组织、保险公司和联邦政府拥有大量关于患者、他们的卫生保健问题、临床治疗过程、他们的费用和结果的数据。理解这些数据之间的联系对于许多问题都是至关重要的,这些问题涉及从确定何种治疗过程和临床方案最有效,到如何在资源减少的年代最好地为大多数人提供卫生保健。

7) 多媒体文档:没有多少人满意今天的 Web 文档检索技术,但是文档的数量和访问这些文档的人数一直在急剧增加。此外,存储包括音频、图像和视频数据在内的多媒体数据正变得越来越容易,但是随着文档量的增加,从文档中提取有意义的信息正变得越来越困难。

8) 数据 Web:今天,Web 主要面向文档和它的多媒体扩展。研究表明,HTML 是支持它的简单、但功能强大的语言。未来,Web 可能对于数据也一样重要。可扩展的标记语言(Extensible Markup Language, XML)是一种新兴的网络环境下处理数据的语言。随着这种基础设施的成熟,对于新兴的数据 Web,数据挖掘可望成为一种至关重要的可行技术。

1.7 影响数据挖掘的趋势

本节介绍五种可能对数据挖掘具有重要影响的发展趋势。

1) 数据发展趋势:或许,最重要的发展趋势是过去 20 年来数字数据的爆炸式增长。在

此期间,数据量可能已经增加了6~10个数量级。大部分数据都可以通过网络访问。另一方面,与此同时,可以分析这些数据的科学家、工程师和其他分析人员的数量保持相对稳定。例如,在此期间,每年毕业的统计学博士数量保持相对稳定。唯一可能的结论是:

要么大部分数据注定是只写的,要么必须开发像数据挖掘这样的技术,可以(部分)自动地分析这些数据,过滤不相关的信息,并提取有意义的知识。

2) **硬件发展趋势:**数据挖掘需要在大型数据集上进行数值和统计密集的计算。工作站内存的增加和处理速度的提高使得可以使用当前的算法和技术挖掘几年前因为太大而无法挖掘的数据集。此外, SMP 工作站和高性能工作站集群的高性能计算的装备使得我们可以着手解决几年前只能由最大的超级计算机才能处理的数据挖掘问题。

3) **网络发展趋势:**下一代因特网(NGI)将以 OC-3 (155Mbps) 或更快的速度连接站点。这比当前网络提供的连接快 100 倍。有了这种连接,使用当前的算法和技术处理分布式数据集将成为可能。此外,还正在开发新的协议、算法和语言,以便使用当前和下一代网络进行分布式数据挖掘。

4) **科学计算发展趋势:**正如前面所提到的,今天的科学家和工程师将模拟看作一种科学形态。数据挖掘和知识发现在联系理论、实验和模拟这三种科学形态中扮演了重要角色,特别是当实验或模拟导致大型数据集时尤其如此。

5) **商业发展趋势:**与以前相比,今天的商业活动必须获得更大利润、具有更快的反应、提供更高质量的服务,并且使用更少的人员、更低的成本。在这些期望和约束下,数据挖掘成为一种基本技术,使得商业机构能够更准确地预测顾客和顾客交易带来的机遇和风险。

1.8 研究挑战

在本节中,我们介绍三个研讨会提出的一些研究挑战。这些研究挑战分成五大领域:①提高数据挖掘算法的可伸缩性,②挖掘非向量数据,③挖掘分布式数据,④提高数据挖掘系统和环境的易用性,⑤数据挖掘的隐私和安全问题。

1) **数据挖掘算法的可伸缩性:**今天的大部分数据挖掘算法都假定数据可以放在内存中。尽管常常声称成功地用于大型数据集,但是这些通常只是对大数据集抽样,直到数据集可以放在内存中为止。基本挑战是使数据挖掘算法具有可伸缩性,原因是

- 记录或观测数据的增加。
- 每个观测数据的属性数目增加。
- 用于分析观测数据集的预测模型或规则集的数目增加。
- 交互和实时响应要求的增加。

不仅需要开发当前数据挖掘算法的分布式、并行和非内存版本,而且需要开发真正的新算法。例如,今天的关联规则挖掘算法可以用一、两遍扫描分析非内存数据,而只需要在内存中保留一些辅助数据。

2) **扩展数据挖掘算法到新的数据类型:**今天,大部分数据挖掘算法在向量值数据上运行。一个重要的挑战是扩展数据挖掘算法使之在其他数据类型上运行,包括①时间序列和过程数据;②非结构数据,如文本;③半结构化数据,如 HTML 和 XML 文档;④多媒体和协同数据;⑤层次和多标度数据;⑥集合值数据。

3) **开发分布式数据挖掘算法:**今天,大部分数据挖掘算法需要将所有待挖掘的数据集中到单个、集中的数据仓库中。一个基本挑战是开发数据挖掘算法的分布式版本,使得数据挖掘可以分布地进行,而让某些数据留在原地。此外,为了挖掘分布式数据,需要合适的协

议、语言和网络服务来处理挖掘分布式数据所需要的元数据和映射。随着无线计算和普适计算环境越来越普遍,还必须开发挖掘这些类型的系统所产生的数据的算法和系统。

4) 易于使用: 今天的数据挖掘最多是一个半自动的过程, 并且今后也许仍然如此。另一方面, 一个基本挑战是开发即便对于偶然用户也比较容易使用数据挖掘系统。相关技术包括改进用户界面, 支持大型分布式数据集的偶然浏览和可视化, 开发管理数据挖掘所需要的元数据的技术和系统, 开发合适的语言和协议提供对数据的偶然访问。此外, 另一个重要的基本挑战是开发数据挖掘和知识发现环境, 处理数据收集、处理、挖掘和可视化过程, 以及处理数据和导出信息所需要的协同和报告。

5) 隐私和安全: 数据挖掘是从数据中提取有用信息的有力工具。随着可用的数字数据的增加, 数据挖掘滥用的可能性也在增长。一个基本挑战是为数据挖掘开发隐私和安全模型以及合适的协议, 并确保下一代数据挖掘系统完全使用这些模型和协议进行设计。

1.9 实验平台和基础设施

实验研究在推动数据挖掘领域向前发展的过程中起着至关重要的作用。为高性能和分布式数据挖掘开发的实验平台对于推动该领域的进展是至关重要的。

数据挖掘对实验平台的要求不同于通用的高性能计算平台。例如, 数据挖掘实验平台既是面向处理机的, 也是面向磁盘的; 网络资源必须以确保质量的服务在地理分布的站点之间移动数据集和数据元素; 必须有各种通用和专用的数据挖掘软件。

或许, 创建数据挖掘实验平台和国家级数据挖掘资源的两个最大的挑战是: ①收集合适的数据集, ②需要交叉学科和多学科团队。

参考文献

- [1] M. Goebel and L. Gruenwald, A survey of data mining and knowledge discovery software tools, *SIKDD Explorations*, vol. 1, Issue 1, pp. 22–33, June 1999.
- [2] Michael Lesk, How Much Information Is There In the World?, <http://www.lesk.com/mlesk/ksg97/ksg.html>.
- [3] P. Adriaans and D. Zantinge, *Data Mining*, Addison-Wesley, 1996.
- [4] Elert, Glenn, The Physics Factbook, <http://www.hypertextbook.com/facts/2000>.
- [5] Claude Sammut, Scott Hurst, Dana Kedzier, Donald Michie, Learning to fly, *Proceedings of the Ninth International Workshop on Machine Learning*, pp. 385–393, Aberdeen, Scotland, United Kingdom, July 1992.
- [6] A. Danyluk, F. Provost and B. Carr, Telecommunications network diagnosis, In Willi Klossgen and Jan Zytkow (Ed.), *Handbook of Data Mining and Knowledge Discovery*, Oxford University Press, 2002.
- [7] Pat Langley and Herbert A. Simon, Applications of machine learning and rule induction, *Communications of the ACM*, 38(11), pp. 54–64, 1995.
- [8] Bob Evans and Doug Fisher, Using decision tree induction to minimize process delays in the printing industry, In Willi Klossgen and Jan Zytkow (Ed.), *Handbook of Data Mining and Knowledge Discovery*, Oxford University Press, 2002.
- [9] Claire D'Este, Mark O'Sullivan, Nicholas Hannah, Behavioural cloning and robot control,

Robotics and Applications, 179–182, 2003.

- [10] R.D. King, S.H. Muggleton, A. Srinivasan and M.J.E. Sternberg, Structure activity relationships derived by machine learning: The use of atoms and their bond connectivities to predict mutagenicity using inductive logic programming, *Proc. Nat. Acad. Sci. U.S.A.*, 93, pp. 438–442, 1996.
- [11] Robert C. Holte, Lessons learned from applications of machine learning, *First International Workshop on Data Mining Lessons Learned*, DMLL – 2002, held in The University of New South Wales, Sydney, Australia.
- [12] Berzal, Fernando, Cubero, Juan-Carlos, Marin Nicolas, Serrano, Jose-Maria, TBAR: An Efficient method for association rule mining in relational databases, *Knowledge Engineering*, 37, pp. 47–64, 2001.

第2章 从商务角度看数据挖掘

2.1 引言

信息技术发展的一个新的重要趋势是识别信息系统中有意义的信息。这种知识的获得可能是在业界获得竞争优势的关键。这就需要工具,以便对这些信息进行分析 and 建模。数据挖掘的价值在于主动搜寻产业发展趋势,并将这种理解提供给维护大量信息的组织机构。这样,商务机构的数据挖掘目标主要是改善组织机构与它们的顾客之间沟通的质量。

在早期,由于已有的信息系统缺乏存储和分析信息的能力,公司处于不利地位。然而,从过去和现在的信息中提取模式并预测未来的科学技术今天已经成熟,并且能够为组织机构的各个部门提供极为重要的信息。对于某些公司,数据挖掘意味在人的控制下,使用算法从信息中提取模式。大规模的自动搜索和对发现规律的解解释属于数据库中知识发现(KDD)的范畴。KDD 关注应用于数据库的知识发现过程。KDD 处理存在于所有的科学领域和诸如营销、规划和控制等应用领域的已有信息。通常, KDD 必须处理不确定的信息、含噪声的信息和稀疏的信息。这样, KDD 是指从信息中发现有用信息的整个过程,而数据挖掘是指从信息中提取模式的算法应用。然而, KDD 和数据挖掘差别正在逐渐消失,因此本书将互换地使用这些术语。

如果正确地进行,数据挖掘可以为组织机构提供一种优化其商务信息处理的方法。当前,新的数据挖掘公司正在迅速涌现,以应对提供这种服务的挑战。尽管数据挖掘改善了使用数据挖掘的商务组织与其顾客之间的沟通,但是还有许多数据挖掘公司正在试图纵向联合从生产到销售的各个阶段,为广阔的市场提供最好的服务。这件事正在通过关注特定的行业,并试图理解该行业的信息所收集的信息类型进行着。这样,数据挖掘是从大型数据库中提取有效的、先前未知的信息,并使用它制定至关重要的商务决策的过程。大型复杂数据集的数据挖掘或探索式数据分析结合了统计学和机器学习的知识和研究成果,以便在超大型数据库中发现新知识。

在过去的 30 年中,日益增长的大量关键商务信息已经以电子方式存储,并且信息量将继续增长。尽管收集的商务信息量不断增长,但是信息的价值很少被完全利用。这是因为完全分析这些信息并洞悉可能出现的模式是一项困难的任务。下面给出一个数据挖掘试图解决的这种困难问题的一个例子。像沃尔玛这样的零售公司收集了每位购买者的大量信息。假设沃尔玛想研究库存管理问题。对于销售数以百万计商品的大客户,预测最优库存不是一件容易的事。有许多子问题都非常复杂,需要大量时间来解决。一个这样的子问题是理解沃尔玛的顾客和预测顾客的爱好。在这个例子中,可以使用数据挖掘工具洞察顾客的行为模式,帮助沃尔玛保持合适的库存量。由于一个公司可能保有数十亿或数兆兆字节信息,数据挖掘可以探查这些信息,挑选出所有重要信息,并将它提供给 CEO,以便他更好地理解顾客的交易结构。

商务数据挖掘的演变概括在表 2-1 中。数据挖掘技术是长期研究和产品开发过程的产物。这种演变始于商务信息第一次存入计算机,不断改进信息访问,如最近产生了允许用户

实时地在他们的数据中导航的技术。数据挖掘已经可以用于商务应用，因为它被三种成熟技术所支持：

- 大量数据收集。
- 功能强大的多处理器计算机。
- 数据挖掘算法。

表 2-1 数据挖掘演变

演变步骤	商务问题	可用技术	产品提供者	特 性
数据收集(20 世纪 80 年代)	“我过去 5 年的总收入是多少?”	计 算 机、磁 带、磁盘	IBM、CDC	回顾的、静态的数据传递
数据访问(20 世纪 80 年代)	“新英格兰三月份的单位销售是多少?”	关系数据库(RD-BMS)、结构化查询语言(SQL)、ODBC	Oracle、Sybase、Informix、IBM、Microsoft	回顾地、记录级动态的数据传递
数据仓库与决策支持(20 世纪 80 年代)	“新英格兰三月份的单位销售是多少?” “下钻到波士顿。”	联机分析处理(OLAP)、多维数据库、数据仓库	Pilot、Comshare Arbor、Cognos、Microstrategy	回顾的、多级动态的数据传递
数据挖掘(今天)	“下个月波士顿的单位销售会是什么样?为什么?”	高级算法、多处理器计算机、海量数据库	Pilot、Lockheed IBM、SGI、大量新出现的公司(新生产业)	预期的、前瞻的信息传递

商业数据库正在以史无前例的速度增长。与之相伴的对改进计算引擎的需求现在能够用并行多处理器计算机技术以合理的性价比满足。数据挖掘算法体现了已经至少存在 20 ~ 30 年的技术，但是在最近才作为成熟的、可靠的、可理解的、超越较老的统计学方法的工具实现。在从商务数据到商务信息的演变过程中，每一步都建立在前一步的基础上。例如，动态数据访问对于数据导航应用的钻取是至关重要的，并且存储大型数据库的能力对于数据挖掘是至关重要的。从用户的观点来说，表 2-1 列举的 4 步都是革命性的，因为它们允许准确、快速地回答新的商务查询。数据挖掘技术的核心组成部分已经在统计学、人工智能和机器学习等研究领域发展了几十年。今天，这些技术的成熟，加上高性能的关系数据库引擎和广泛的数据集成，使得这些技术在当前的数据仓库环境下成为切实可行的技术。

数据挖掘产业的一个持续趋势是企业资源规划(ERP)零售商和应用服务提供者(ASP)的出现。许多大型公司都通过实现 ERP 系统而获益。ERP 系统试图将整个公司的所有部门和职能集成到一个计算机系统中，为这些不同部门的特定需求提供服务。另一方面，ASP 旨在提供与 ERP 零售类似的服务，但面向较小的公司，帮助这些公司提升他们的数据管理能力。这两种类型的公司(ERP 和 ASP 公司)都可以借助于提供附加的数据挖掘服务，在市场上赢得更高的地位。提供集成公司的多个部门的已有数据的软件工具，再加上提供与该软件包一起最有效运行的数据挖掘工具，可能为客户公司带来显著利益。

2.2 从数据挖掘工具到解决方案

现在，数据挖掘已经成为商务和软件杂志中许多文章的主题。然而，几年前还没有多少人听说过数据挖掘这个术语。尽管数据挖掘是一个具有很长历史的领域的演变，但是该术语本身的引进相对较晚，大约在 20 世纪 90 年代提出。

数据挖掘可以沿三条谱线追溯。其中最长的谱线是经典统计学。没有统计学就没有数据挖掘，因为统计学是数据挖掘建立于其上的大部分技术的基础。经典统计学包括诸如

回归分析、标准正态分布、标准差、标准方差、聚类分析和置信区间等概念,这些概念都主要用于研究数据和数据之间的联系。这些都是最基本的构件块,可以基于它们建立更高级的统计分析。即使在今天的数据挖掘工具和知识发现技术中,经典的统计学分析仍然扮演重要角色。

数据挖掘的次长谱线是人工智能(AI)。与统计学相反,该学科建立在启发式方法的基础上,试图用类似于人的思考方法处理统计学问题。由于这种方法对计算机处理能力要求极高,直到20世纪80年代初,计算机开始以合理的价格提供相应的功能时,这种方法才变得切实可行。AI在高端科技和政府的应用不太多,并且需要超级计算机,这使得一般人难以应用AI。值得注意的一个例外是某些AI概念被某些高端商品化产品(如关系数据库管理系统的查询优化模块)采用。随着时间的流逝,这些已经改变,AI被用来创建处理和解决复杂的、数学驱动的问题的新方法。MIT的人工智能实验室(创建于20世纪60年代)对智能的许多方面进行着广泛研究。他们的目标有两个:在所有层面理解人类智能,包括推理、理解、语言、发展、学习和社会层面;基于智能构建有用的制品。

数据挖掘的第三条谱线是机器学习,它可以更准确地被描述为统计学和AI的结合。尽管AI并未获得商业成功,因此主要用作一种研究工具,但是它的技术在很大程度上被机器学习吸纳。因为项目价格比AI低,机器学习能够利用20世纪80年代和90年代计算机不断提高了的性价比,找到更多应用。机器学习可以看作AI的演变,因为它融合了AI的启发式方法和高级统计分析。机器学习试图让计算程序学习它研究的数据,使得程序基于所研究的数据的特征做出不同的决策,使用统计学的基本概念,加上更高级的AI启发式方法和算法来实现它的目标。

这样,在许多方面,数据挖掘基本上是将机器学习用于商务和科学应用。最好将数据挖掘看作统计学、AI和机器学习过去和现在发展的融合。这些技术一起使用,研究数据并发现数据中先前隐藏的趋势或模式。在需要分析大量数据以发现其他方法不能发现的趋势的科学和商务领域,数据挖掘正在逐渐被接受。

2.3 数据挖掘系统的演变

第一代:现在称作数据挖掘系统(data mining system)的第一代系统出现于20世纪80年代,主要由关注单一任务的、研究驱动的工具组成。那时,不太需要完全理解数据的多维层面,因为使用一维分析工具就能完成任务。这个因特网出现之前的时代主要分析大型数据库中的单一问题。这些任务包括使用决策树或神经网络工具建立分类器,发现数据中的聚类 and 实现数据可视化。这些工具处理一般数据分析问题,并且它们的用户需要具有复杂的技术,以便理解和解释结果。此外,使用多种工具非常复杂,并且涉及大量数据和元数据变换,即便对于专家而言,这也不是一项容易完成的任务。

第二代:大约在1995年,数据挖掘零售商开发了称作套装(suite)的第二代数据挖掘系统。第二代工具的产生主要源于人们认识到知识发现过程需要各种类型的数据分析,大部分时间都花在数据清理和预处理上。这种发现过程通常包括发现数据中的模式。诸如SPSS的Clementine、Silicon Graphics的Mineset和IBM的Intelligent Miner这样的套装允许用户执行多种发现任务(通常有分类、聚类和可视化),并支持数据变换和可视化。

第三代:尽管第二代系统试图解决商务用户不能直接使用系统的问题,但是这些问题依然存在。它们需要大量的统计学理论,以便支持多种发现任务。这意味为了提取隐藏的信息模式,需要花费大量时间搞清楚应当使用什么算法和如何使用它来产生有用的结果。于是,

就出现了第三代系统。由于商务用户的需要,出现了第三代系统,即20世纪90年代的基于应用和解决方案的从生产到销售全过程的数据挖掘。这些工具主要源于解决具体的商务问题,如预测未来顾客会购买什么或某企业的库存优化。这种知识发现过程筛选存储在大型数据库中的信息,发现隐藏的模式。结果传递给诸如决策支持系统这样的前端应用,使得商务用户能够基于数据挖掘工具所要处理的特定问题进行决策,而忽略数据挖掘工具的细节。

从生产到销售全过程的数据挖掘应用已被开发出来,能为正确的决策提供高回报。这些应用通常解决最常见,同时对于管理者来说也是最为关键的商务问题。这意味着,尽管结果源于应用,但是理解这些数据,并且基于这些数据做出重要和关键的商务决策还取决于企业的经理。此外,随着因特网不断改变企业与竞争对手的竞争方式,数据挖掘为企业提供了新的竞争潜力。现在,这些拥有海量信息的企业可以评估数据挖掘工具的回报。这些积累了几个数量级数据的企业可以将数据挖掘工具用于这些数据,帮助企业做出最佳决策。

因特网充斥许多处于数据库存储前沿的新的数据类型。随着数量的增长,数字图书馆正用来存储诸如声音、文本、视频和图像等数据。Web数据挖掘或Web挖掘对数据挖掘公司提出了新的挑战。分析顾客日志中的点击流数据,为特定的顾客实时地确定正确的广告或推荐产品是数据挖掘公司面临的新问题。为了实时地确定顾客应当有什么样的弹出菜单,出现了多种技术来提供上下文敏感的菜单,从而为公司提供这种服务。研究者正在开发新的方法,为Web站点做预测。协同过滤最初是在MIT开发的,并在Firefly Network(萤火虫网络)(1998年被微软收购)和Net Perception(网络感知)中实现,现在用于网络站点,试图预测顾客未来的购买模式。协同过滤基于这样的假定:寻找信息的人应当能够利用其他人已经发现和评估的信息。这样,这些系统根据顾客购买和选择的商品信息,预测顾客将来可能的购买行为。当前的协同过滤系统为读者提供工具,基于读者组群的凝聚评估过滤文档。

2.4 知识发现过程

知识发现过程或数据挖掘的步骤概述如下:

- 定义问题。这个初始步骤涉及理解问题和了解项目的目标和期望是什么。
- 收集、清理和准备数据。这需要了解需要什么数据、哪些数据是最重要的并集成信息。完成这一步需要很大的工作量,大约占整个数据挖掘工作量的70%。
- 数据挖掘。这个模型构建步骤涉及选择数据挖掘工具,如果数据挖掘工具需要的话,还要变换数据,产生训练和检验模型的样本,并且最终使用该工具构建和选择一个模型。
- 验证模型。检验模型,确保它产生正确和足以满足需要的结果。
- 监控模型。监控模型是必要的,因为随着时间流逝,必须重新验证模型,以确保它仍然满足要求。一个今天运行良好的模型明天可能就无法满足要求,因此必须监控模型的行为,确保它满足性能要求。

2.5 数据挖掘支撑技术概述

数据挖掘是多种技术的集成,如图2-1所示。统计学、决策支持系统、数据库管理和数据仓库、机器学习、可视化和并行处理技术都是相互影响和支持数据挖掘工具的工具。统计学和机器学习都继续朝着更复杂的统计技术发展。决策支持系统是一组工具和过程,帮助经理做出决策并指导他们进行管理。例如,用来安排会议、组织活动的工具,电子数据表图形工具和性能评估工具都是支持系统的例子。可视化技术用来支持数据挖掘过程。来自计算机

可视化领域的研究者以不同的方式进入数据挖掘领域：为数据挖掘者提供交互式的数据挖掘工具。数据库管理和数据仓库集成各种数据源，组织数据使得数据可以有效地挖掘，为数据挖掘过程提供支持。最后，数据挖掘算法的可伸缩性是需要关注的问题之一。使用并行处理的技术是数据挖掘进展的另一个关键技术支撑。

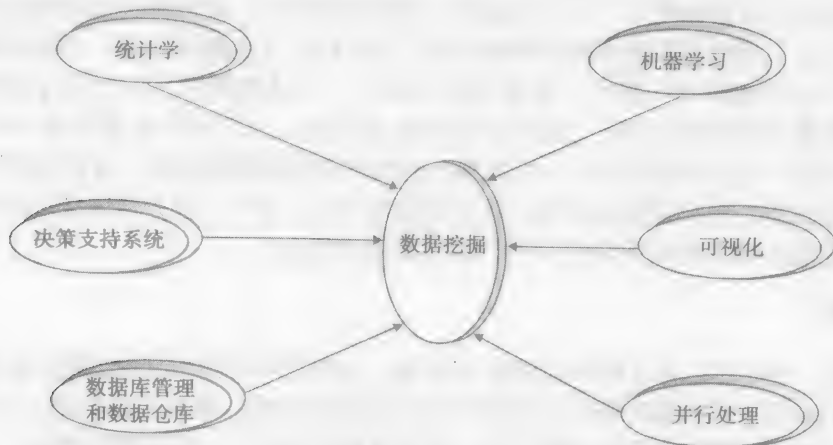


图 2-1 数据挖掘技术

2.5.1 数据挖掘：验证与发现

数据挖掘不是简单的查询提取、验证/分析工具。决策支持系统(DSS)、主管信息系统和查询生成工具主要用来产生关于数据的报表。以这种方式，这些查询工具主要用来访问已经存放在大型数据库中的记录。提取数据之后，可以检查这些数据以获得已有模式或用来回答感兴趣问题的其他信息。这种发现已有数据中的趋势的数据提取方法称作验证方法(verification method)。在这种模式下，数据挖掘者必须对感兴趣信息的存在性做出假设，将假设转换成查询提交给数据仓库，并针对所做的决策解释返回的结果。使用这种方法将不断地提出假设，使用查询工具支持或否定它。这是一种为提出的问题找出一种可能解的系统方法。使用这种检索方法并不创建多少信息。

数据挖掘使用的方法与此不同。数据挖掘使用发现方法，根据所使用的方法，该技术只凭借少量用户指导，试图发现趋势并产生关于数据的结果。这种技术用来发现最重要的数据源，并根据筛选的数据得出结论。数据挖掘考察大量数据，发现事实并提醒进行挖掘的人注意。就这一点而论，数据挖掘是发现工具，它实现更有效的探索相关数据的方式并将这些信息提供给数据挖掘者。

2.5.2 决策支持系统

决策支持系统(DSS)通常与数据挖掘工具相关联，帮助主管做出更有远见的决策(见图 2-1)。尽管当前市场上存在形形色色的决策支持系统，但是它们的应用主要是为主管综合数据，使得他们能够基于数据分析做出更客观的决策。DSS 技术产生于 20 世纪 80 年代中期，并且 DSS 分析工具也在不断改进。在当今 Internet 时代，联机分析处理(OLAP)正慢慢地取代老化的决策系统。从本质上讲，如果一个计算机化系统不是联机事务处理(OLTP)系统，则该系统被看作是 DSS。逐渐地，OLAP 和多维数据分析用于决策支持系统，以便从数据库

中发现信息。主管信息服务(EIS)、地理信息系统(GIS)、OLAP和知识发现系统都可以划归到DSS的系统范畴。DSS的两个主要类别包括企业范围DSS和桌面DSS。在企业范围DSS中,DSS连接到大型数据仓库,通常为组织内部的许多经理提供服务。这种大型基础设施使经理们能够快速访问数据。在企业范围DSS中,最复杂的企业范围分析系统提供对一系列面向决策的数据库或数据集市、预定义的模式和图表的访问,提供对公司数据仓库中变量的即时访问。此外,像数据挖掘这样的数据分析工具可以进一步操纵数据,帮助经理做出有远见的决策。与企业范围DSS相反,桌面DSS主要由一个经理使用。在大部分组织中,企业范围DSS、数据仓库和桌面DSS之间存在持续的通信流。企业可以实现各种DSS结构。例如,可以实现单一的企业范围DSS,以处理企业的所有数据和操作流;或者用其他决策支持系统,如驻留在单个用户桌面的桌面DSS实现多层DSS。客户-服务器体系结构可以在客户桌面和驻留在服务器上的相关联的DSS工具之间传递信息。

2.5.3 OLAP

OLAP是一种技术,可以根据存储在关系和二维数据库表中的信息构建多维数据立方体。用户可以使用这些数据回答现实世界中复杂的商务查询。联机分析处理(OLAP)是一种软件技术,能够快速分析共享的多维信息。多维数据是用以下方式组织的数据:允许同时在多个维上查看和比较数据——与电子数据表的二维(水平和垂直)结构截然不同。尽管电子数据表允许用户比较二维数据,但是多维结构提供几乎无限多个视图和关联。为了快速地回答诸如“北印度2003年第四季度销售最好、至少具有20%利润空间并且通过间接销售渠道销售的生产线是什么?”之类的复杂问题,多维分析是必要的。

OLAP通过为商务用户提供快速的、不受限制的大量汇总数据的多重联系视图,使得用户能够做出更好的决策。这可能导致对大量汇总数据多维分析的高性能访问。使用OLAP,经理和分析人员能够快速、容易地在海量数据上考察关键性能数据,进行比较和趋势分析。数据比较可以用于各种商务领域,包括营销分析、财务报告、质量跟踪、赢利分析、人力和定价应用,等等。OLAP使用数据仓库技术,根据来自企业范围的数据创建信息库(见图2-2)。无论数据驻留何处,对于网络任何位置上任何支持平台的应用请求,包括基于Web的应用,数据都是可访问的。

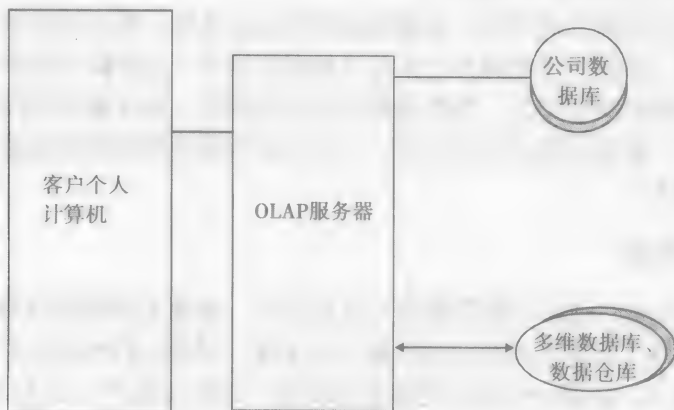


图 2-2 OLAP 概览

2.5.4 桌面 DSS

桌面单用户的 DSS 不如企业范围的系统流行,因为它们不允许多个用户链接到大型数据仓库获得信息。然而,桌面 DSS 确实是有用的。个人用户可以使用微软 Excel、Lotus 1-2-3 或其他专门的 DSS 应用程序为某个经理分析信息。Expert-Choice 就是用作桌面 DSS 的专门 Windows 软件包的例子。ExpertChoice 是一种软件工具,主要通过分析由目标、可能的场景、标准和选择组成的层次模型,从而支持决策。例如,一个使用 Excel 对财务问题建模的 Goldman Sachs 的分析人员可以将他/她的发现作为企业范围 DSS 的程序组件提供给经理。然后,分析人员可以继续分析,并且一旦完成就可以把他们的发现放到公司内部的互联网上。

一般来说,无论是解决战略决策还是运营决策,DSS 都具有明确的用途。可以用来检索和分析数据的工具、数据库、包括的变量和数据的时间序列决定可以提出的问题和可以产生的决策相关的信息。然而,从长远来看,DSS 可以帮助经理提取、汇总和分析决策相关的数据,制定更加明智和有远见的决策。

2.5.5 数据仓库

数据挖掘和 DSS 是两类使用数据仓库的应用。通过 DSS,一个经理可以做出更有远见的决策。然而,通过与数据仓库相结合,DSS 可以更好地使用高质量的信息,并以更有效的方式解释信息。从本质上讲,数据仓库是一个企业或组织的商务系统收集的所有或重要数据的中心存储[1, 2, 3]。通常,数据仓库存放在企业的大型服务器上。来自各种联机事务处理(OLTP)应用和其他信息源的数据有选择地被提取和组织到数据仓库的数据库中,用于诸如数据挖掘工具这样的分析应用(见图 2-3)。这样,通过提供进行分析的集成的、历史数据平台,它用于支持需要数据管理的组织的信息处理。

数据仓库的 4 个主要特征:数据仓库包括如下 4 个主要特征:

- 面向主题的
- 集成的
- 时变的
- 非易变的

数据从操作环境进入数据仓库(见图 2-3)。也就是说,数据仓库逻辑上总是数据的独立存储,这些数据是从操作环境得到的应用数据的变换而来的。数据仓库的一个主要特征是它是面向主题的。作为一个例子,数据仓库可以围绕企业的顾客、产品或研发进行构建。相比之下,操作数据库系统多半围绕主题领域的应用和功能进行组织。这两种系统结构的主要区别是数据仓库排除不被 DSS 工具使用的所有信息。然而,操作数据库系统包含所有数据,无论 DSS 工具是否使用这些数据。

数据仓库的另一个特点是数据仓库中的数据是一致的。例如,数据仓库使用一种特定日期格式,而不是多种日期格式。这种一致性适用于进入数据仓库的所有数据。所有的变量、命名方案和编码结构都遵循预先的约定。对于研究数据的 DSS 分析人员,他/她的关注点将

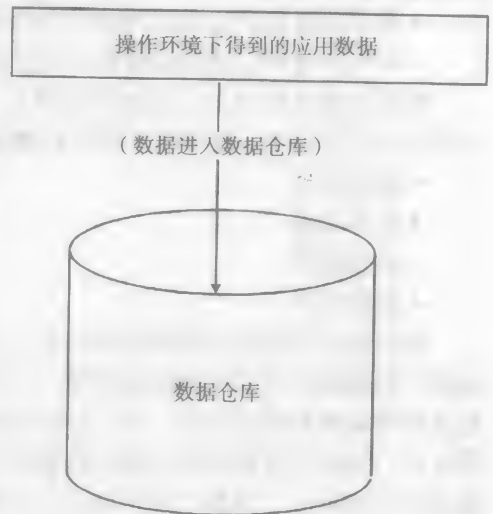


图 2-3 数据仓库

是使用数据仓库中的数据,而不必担心数据的一致性和可信性。

数据仓库的第三个特征是时变性。在操作环境下,希望数据在被访问时是精确的。然而,在数据仓库中,数据在某时间段是精确的。其主要区别是操作环境与仓库环境的时间范围显著不同。在数据仓库中,数据长时间保存,通常要保留若干年。相比之下,在操作环境下,数据通常维护数月。这样,操作环境下的应用必须具有高度的灵活性,因为数据会不断地更新。在数据仓库环境下,维护大量记录会给更新数据仓库带来困难。

最后,数据仓库的第四个有趣特征是它是非易变的。尽管在操作环境下经常更新,但是在数据仓库中,一旦数据装入就不会出现对数据的进一步更新。不需要像操作环境那样,通过备份与恢复来维护记录,也不再需要事务、数据完整性和死锁检测。

这样,这四个基本性质使得数据仓库的环境明显有别于操作环境。然而,进入数据仓库的所有数据都来源于操作环境。数据仓库充当一种工具,把这些数据变换成分析与综合数据时更有用的形式。数据仓库的维护和设计的简洁性使得数据仓库成为向数据挖掘工具传送高质量信息的重要因素。面向主题的、集成的、时变的和非易变的特性对于为其他系统维护一致数据都起到重要作用。

2.5.6 数据挖掘过程

识别和利用隐藏在数据中信息的目标有三个要求:

- 捕获的数据必须集成到企业范围的视图,而不是特定的视图。
- 必须提取包含在集成的数据中的信息。
- 必须以有利于制定决策的方式组织得到的信息。

数据挖掘过程[4, 5]可以分为四步,将数据仓库中已经汇总的数据可以转换成能够产生有用结果的信息。这四个步骤可以概括为:

- 数据选择
- 数据变换
- 挖掘数据
- 解释结果

数据选择为进行分析而收集数据。数据变换(细节见第4章)将数据变换成特定的格式。数据挖掘将提取所期望的信息类型,产生待解释的结果。在图2-4中,数据挖掘工具将从数据仓库环境提取相关信息。为了运行数据挖掘工具,必须在数据挖掘之前进行数据选择和数据变换。结果传递到面向决策的数据库或数据集市,在那里用户可以根据结果提出建议,并将建议付诸行动。当然,假定这四个步骤都能成功完成,但事实并非总是如此。

数据选择可能是这一过程中最重要的步骤。这是因为在数据提取实际进行之前预先找出和构造选择标准非常复杂。这一步应当确定选取的变量和它们的值域。例如,一位希望提高销售量的销售主管将预先选择购买行为非常活跃的顾客并观察他们的行为。该主管可以挖掘所有的数据,但这样做可能成本非常高,因为数据挖掘工具必须搜索所有这些数据,并且得到结果后在预测最佳推荐时具有更大的风险。因此,小心地选择数据是一个非常重要的步骤。

一旦选择了待挖掘的数据,数据挖掘过程的下一步就是将数据变换成数据挖掘工具所要求的特定格式。通过一些变换并应用算法将数据转换成适合进一步使用数据挖掘工具处理的特定格式,数据被进一步综合。

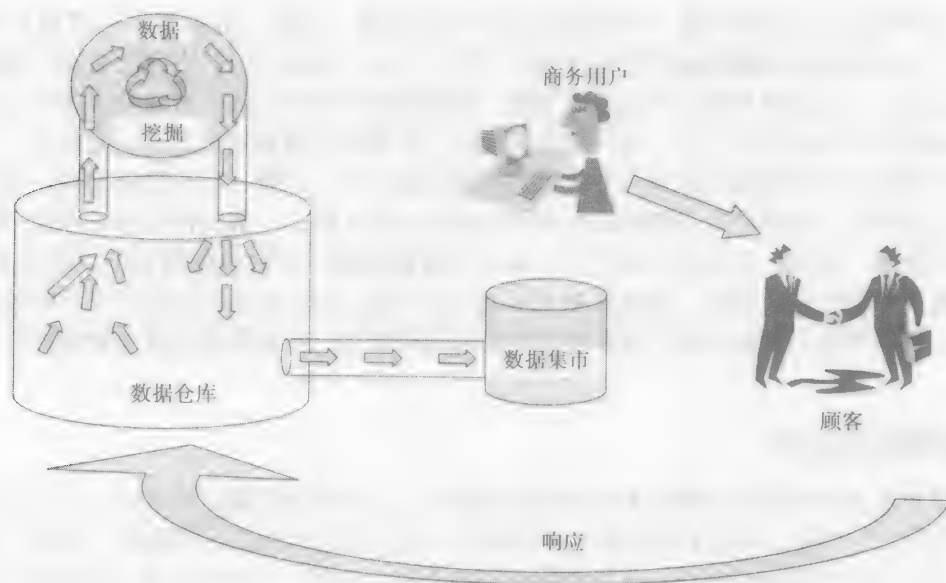


图 2-4 数据挖掘过程

一旦完成数据选择和所需要的变换，就可以使用数据挖掘工具。通过使用设计良好的算法（稍后章节讨论），基于先前收集的数据对未来事件的特定预测可以产生重要的发现。建议数据仓库与数据挖掘工具一起使用，因为这可以更有效地以有利于分析的方式组织数据。此外，数据挖掘工具也可以与 DSS 连接，以便进一步解释数据。然而，数据挖掘系统不必总是与数据仓库交互，并且事实上数据挖掘还可以从数据库中的原始数据中提取相关信息。使用数据仓库的主要优点是：大部分数据已经集成为合适的格式，使得数据挖掘工具更容易提取高质量的信息。

数据挖掘过程的最后一步是解释结果。一旦分析和解释了所提取的信息，就可以将最相关的信息通过 DSS 传递给决策制定者。结果的解释不仅包括解释输出，而且包括进一步过滤数据并将信息传递到决策支持系统。当解释的结果不令人满意时，就要重复先前的步骤，直到所产生的信息对于数据挖掘者而言具有最大的附加价值为止。

可见，数据挖掘是一个非常复杂的过程。在将数据提供给数据挖掘工具之前，许多步骤都需要正确地执行。此外，不能确保数据挖掘工具在挖掘过程任何一步都产生有意义的结果。毫无疑问，应当进行一些试验，因为试验可以揭示每个步骤的误差校正。可以修改前面提到的每一步来进一步考察数据，搜索隐藏的模式。这是数据挖掘组织所面临的挑战，尽管这可能是一个需要付出极大努力的过程。被挖掘的数据越多，挖掘者从这一过程中学到的也越多。

像 DSS 和数据仓库环境这样的工具的使用完善了用来发现隐藏在数据层中的有用事实的数据挖掘工具。为了尽可能提高数据挖掘的效率，这些工具都必须向数据挖掘工具提供高质量的信息传递。过滤数据的辅助工具的使用与功能强大的数据挖掘工具一起应当是设计良好的环境的一部分。

DSS 和数据仓库与数据挖掘工具集成，提取数据中隐藏模式的方式如图 2-4 所示。图中所显示的是数据挖掘框架提取工具所必需的三个主要成分。正如前面所讨论的，DSS 确实使得经理可以考察数据，帮助他或她做出决策。DSS 将得到从数据挖掘工具传递过来的结果。

数据仓库系统将综合和集成输入到数据挖掘工具的数据。这样,数据挖掘工具将与 DSS 结合,为企业实现数据挖掘策略提供最终解决方案。然而,在此之前,在将数据输入到数据挖掘工具之前,必须集成和预分析选取的数据。集成数据涉及将主要驻留在具有多个文件或数据库的操作环境中的数据合并。进行数据变换的工具通常与数据挖掘工具一起提供。这是因为数据挖掘开发者也构建集成与合并数据挖掘功能的工具。注意,对于数据挖掘,构建数据仓库不是必需的。数据可以直接从操作文件下载到一般文件,一般文件包含可以用于数据挖掘分析的数据。然而,在大部分情况下,输入到数据挖掘工具的数据需要加以综合和集成。这是数据仓库的任务。通常,数据挖掘工具将通过 SQL 接口访问数据仓库综合的数据。数据仓库、数据挖掘工具和决策支持系统之间的通信将继续,直到数据挖掘者找到最佳解决方案为止。

2.6 数据挖掘技术

数据挖掘技术可以解决哪些类型的商务问题?为了有效地使用这些工具,工具的使用者必须理解什么?诸如“产品 X 的销售七月份增长了吗?”或“当产品 Y 促销时,产品 X 的销售减少了吗?”这类问题容易解决,无需数据挖掘的支持。此外,已有的工具,如 OLAP 和统计技术可以用于这种情况,以分析这类问题。相比之下,使用数据挖掘,我们可以问这样的问题:“决定产品 X 销售的最佳因素有哪些?”然而,对于解决某类问题,并非所有的数据挖掘工具都是最好的。对于特定类型的问题,有些工具比其他工具更合适。

使用传统的工具,试图得到诸如上述问题答案的分析者将艰难地试图通过试验产生一个模型。他或她将首先就一个假设提出一系列假定,然后检验它,最后提出附加的假设,并以迭代的方式重复检验过程,建立一个模型。使用数据挖掘,尽管需要做出一系列假设和假定,检验它并进行修订,但是使用数据挖掘工具的优点是发现合适模型的大部分工作从分析者转移到计算机。这样,产生模型需要的工作量大大减少,并且使用计算机可以评估大量模型,增加了找到正确模型的几率。

工具类

- 关联
- 序列模式
- 分类/回归分析
- 决策树
- 神经网络
- 可视化
- 聚类
- 协同过滤
- 数据变换和清理
- 偏差和欺诈检测
- 估计和预报
- 贝叶斯和依赖网络
- OLAP 和 multidimensional 分析
- 统计分析
- 文本分析
- Web 挖掘

以上数据挖掘应用方法的列表包含了数据挖掘使用的大部分技术。当前数据挖掘的比较大的应用主要是关联和序列模式工具、分类、可视化和聚类。协同过滤是主要用于 Web 挖掘的相对较新的应用。

参考文献

- [1] W.H. Inmon, *Building the Data Warehouse*, 2nd ed., John Wiley & Sons, 1996.
- [2] Ralph Kimball and Margy Ross, *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling*, 2nd ed., Wiley, April 12, 2002.
- [3] Paulraj Ponniah, *Data Warehousing Fundamentals, A Comprehensive Guide for IT Professionals*, Wiley Interscience, August 3, 2001.
- [4] Michael J.A. Berry and Gordon S. Linoff, *For Marketings Sales and Customer Relationship Management*, 2nd ed., Wiley Computer Publishing, April 5, 2004.
- [5] Olivia Parr Rud, *Modeling Data for Marketing, Risk and Customer Relationship Management*, Wiley, November 3, 2000.

第3章 数据挖掘算法的数据类型、输入和输出

3.1 引言

现代科学与工程使用基本原理(first-principle)模型描述物理、生物和社会系统。这种方法从像牛顿运动定律或麦克斯韦电磁学方程这样的基本科学模型开始,然后在其上建立各种机械工程或电子工程应用。在这种方法下,实验数据用来验证基本原理模型,估计某些有时很难或不可能直接测量的参数。然而,在许多领域,基本原理是未知的,或者所研究的系统太复杂,很难以数学方式形式化。随着计算机的广泛使用,大量的数据由这种系统产生。在缺乏基本原理模型的情况下,可以使用这种数据,通过估计系统变量之间的有用联系(即未知的输入-输出依赖)导出模型。这样,就存在一个从基于基本原理的建模与分析,到直接由数据开发模型和进行相应分析的范型转移。

在今天的具有大型 Internet 基础设施的基于多媒体的环境下,产生了不同类型的数据并被数字化地存储。为了准备合适的数据挖掘方法,我们必须分析数据集的基本类型和特征。这种分析的第一步是根据数据的计算机表示和使用对其系统地分类。通常作为数据挖掘过程数据源的数据可以分成三类:结构化数据、半结构化数据和无结构数据。大部分商务数据包含结构化数据,它们具有良好定义的字段,取数值或文字值,而科学数据库可能包含所有这三类数据。半结构化数据的例子有商业文档的电子图像、医疗报告、行政报告和维修手册。Web 文档多数都属于这一类。无结构数据的一个例子是商场监控器记录的视频数据。由于硬件价格的下降,感兴趣的事件或过程的可视化的视频和多媒体记录正日趋流行。这种形式的数据一般需要经过大量处理,提取和组织其中的信息。结构化数据通常称为传统数据,而半结构化和无结构数据统称非传统数据(也称多媒体数据)。当前的大部分数据挖掘方法和商品化工具都用于传统数据。然而,应用于非传统数据的数据挖掘工具,以及将非传统数据转换成结构化格式接口的开发正在快速进行中。

3.2 实例和特征

用于数据挖掘的结构化数据的标准模型是案例(实例)的汇集。要指定称作特征(feature)的可能测量值,并且这些特征在许多案例上统一测量。通常,数据挖掘问题的结构化数据表示采用表或单个关系(关系数据库使用的术语)形式,其中列是存放在表中的对象的特征,行是具体实体在这些特征上的值。图 3-1 给出了数据集和它的特征的简化图示。在数据挖掘文献中,我们通常使用术语案例或样本表示行。在数据挖掘中,通常有许多不同类型的特征(属性或变量),即结构化数据记录的不同字段。在处理不同类型的特征方面,并非所有的挖掘方法都一样好。

图 3-2 和表 3-1 用更具体的方法解释实例、特征和结构化数据格式概念。

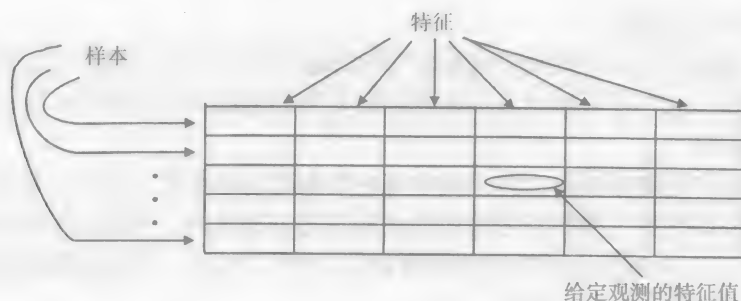


图 3-1 使用特征的对象(样本)的表示

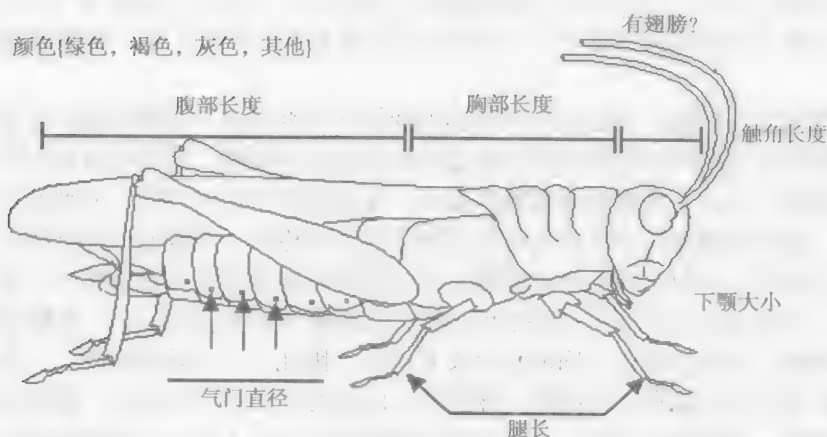


图 3-2 感兴趣域的度量特征

表 3-1 实例、特征和类

昆虫 ID	腹部长度	触角长度	昆虫类
1	2.7	5.5	蝗虫
2	8	9.1	蝸蝓
3	0.9	4.7	蝗虫
4	1.1	3.1	蝗虫
5	5.4	8.5	蝸蝓
6	2.9	1.9	蝗虫
7	6.1	6.6	蝸蝓
8	0.5	1	蝗虫
9	8.3	6.6	蝸蝓
10	8.1	4.7	蝸蝓

3.3 特征(数据)的不同类型

特征有多种类型,可以根据它们编码的信息量加以区分。这里,我们按照从“最简单的”(携带最少信息)特征到携带最多信息的特征的顺序简略回顾它们。

标称变量:本质上,这些只不过是标识唯一实体的标号。人名是识别唯一个体的标称标号。序号、序列号、跟踪码以及其他许多类似的标号均属于此类。

分类变量:这些是组标号,识别共享分类所蕴涵的一些特征的实体组。除人名之外,本

书的所有读者属于人类类别。

序数变量：这些是可以合理地以某种次序列举的类别。这些类别可以包括小、中、大，或者热、暖、温热、凉、冷。注意，标称变量和分类变量都不是有序的，它们只是单个实体和实体组的无序标号。

区间变量：这些是有序变量，可以确定有序类别之间的距离。然而，它们的区间可以是非常任意的，如温度刻度。等距点之间的加法距离是有意义的，但是比例没有意义。例如，20度和30度之间，110度和120度之间的距离都是10度。然而，无论是摄氏温度还是华氏温度，50度都不是25度的2倍那么热。

比例变量：这些是区间变量，其中比例是有效的，并且具有真零点。一个例子是银行账户。零点是空账户。10卢比和20卢比之间的比例是1:2，而20卢比是10卢比的2倍。100卢比和200卢比之间的比例也是1:2，而200卢比是100卢比的2倍。相同的比例，相同的联系。

就数据挖掘算法而论，它们并不区分标称变量和分类变量，尽管挖掘者可能很好地区分了它们。挖掘算法对区间变量和比例变量之间的差别也不敏感，尽管挖掘者也可能对它们之间的区别很敏感。因此，就数据挖掘算法而言，变量或者是无序标号，或者是有序标号，或者是连续数。这意味着算法的敏感性可以分别描述为标称的、序数的或连续数的。如果算法对任何序都不敏感，而只对标号出现的联合频率敏感，则算法是标称敏感的。标号可能实际上是连续数，但是算法只把它们(或更可能是它们的箱)看作标称标号。序数敏感算法对值出现的顺序敏感，但是推导值之间的距离没有意义。因此，对于这种算法，1和2之间存在的间隔与100和200之间存在的间隔一样显著。间隔的大小是不相关的。数值敏感算法对值之间的距离敏感，并且对于它们，无论有没有值落在中间，1和2之间的距离与100和200之间的距离是不同的。

3.4 概念学习与概念描述

数据挖掘的应用有4种类型。

分类学习：这里，学习模式取一个被分类的实例集合，希望由它学习一种对未知实例分类的方法。下面给出一些例子，这些例子曾用于训练鸽子。图3-3和图3-4分别给出分类问题和它的解。

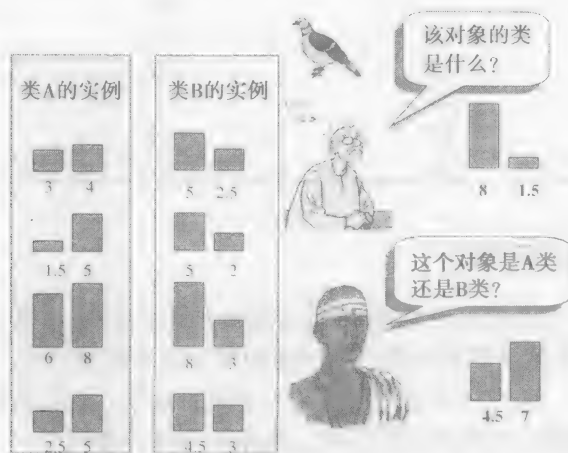


图 3-3 问题 1

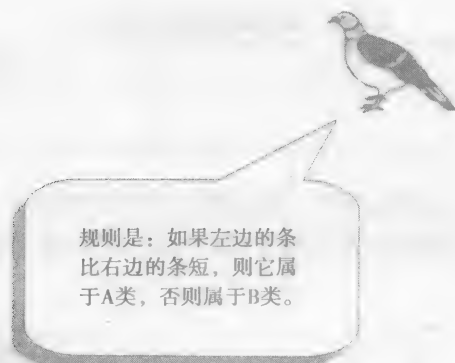


图 3-4 鸽子学习的规则

是的，鸽子是对的！鸽子可以从给它的实例中学习一些概念。现在该你了。发挥你的聪明才智，解决图 3-5 中的问题，这意味比鸽子更聪明。该问题的答案在图右下方给出。

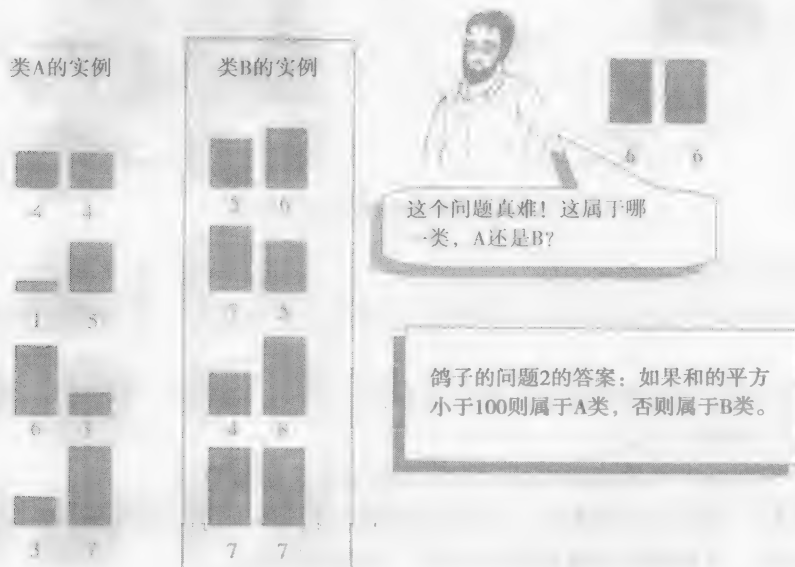


图 3-5 问题 2

一个实际应用。考虑一个为顾客提供贷款的银行。显然，银行发现能够预测哪些新顾客可能是好的投资对象，哪些不是。使用所收集的关于以前顾客的数据，银行想知道那些使得顾客成为好的投资对象或不良投资对象的属性。所需要的是一个规则集，将数据划分成互斥的两组——一组是好的投资对象，另一组是不良投资对象。这种规则称为分类规则 (classification rule)，因为它们将给定的数据分为固定数目的组。老顾客的数据 (他们所属的组是已知的) 称为训练集 (training set)，由它发现规则。之后，用分类规则来发现新顾客属于哪个组。

关联学习：搜索特征之间的任意关联，而不仅仅是预测特定类的特征。例如，考虑如表 3-2 所示的选自历史销售数据的超市销售记录。通常，这种数据的大小从 50KB (研究用) 记录到数 TB。向数据挖掘提出的典型问题是“通常，哪些产品会被一起购买？”答案在数据中，如果我们只能看到数据的话。

表 3-2 超市销售记录的实例

日期和事务 ID	鱼	木豆	大米	葡萄酒
4/3/05 -1	N	Y	Y	N
4/3/05 -2	Y	N	N	Y

聚类：聚类搜索属于同一组的实例。分组基于特征的相似性。根据我们选取的特征（和相似性度量），我们可以把项的集合划分成不同的组群。图 3-6 显示了两个这样的组群（有时可能得到意想不到的结果）。



图 3-6 基于相似性的自然组群

聚类算法可以应用于许多领域，例如：

- 营销：给定包含顾客特征和以前购买记录的大型数据库，发现具有相似行为的顾客组群。
- 生物学：给定动植物的特征，对它们分类。
- 图书馆：给图书分类。
- 保险：识别高于平均索赔率的汽车保险持有者，识别欺诈。
- 城市规划：根据房屋的类型、价值和地理位置，识别房屋的组群。
- 地震研究：对观察到的地震震中聚类，识别危险地区。
- WWW：文档分类，聚类 Web 日志数据，发现相似的访问模式。

数值预测：在数值预测中，预测的输出不是离散类，而是数值量。

无论学习涉及的类型是什么，我们把要学习的内容称作概念，学习模式产生的输出称为概念描述。

3.5 数据挖掘的输出——知识表示

3.5.1 分类学习算法的知识输出

在机器学习中，应用两种方法来学习分类。它们是基于神经网络的方法和基于归纳的方法。两种方法都有各自的优缺点。

决策树：决策树是一种基于知识表示的树，用于表示分类规则。叶节点代表类标号，而其他节点代表与被分类对象相关联的属性。树的每个分支代表对应属性节点的每个可能值。图 3-7 显示了一棵典型的决策树。该树对应于鸢尾植物（Iris）的 3 个物种的分类。分类基于

花瓣宽度和花瓣长度两个属性。更多的细节请参见第6章。

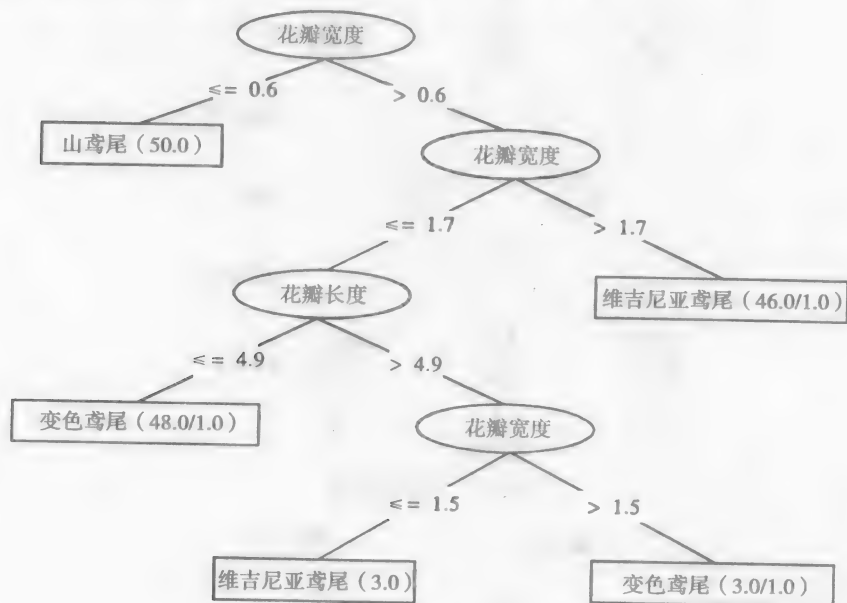


图 3-7 鸢尾植物数据集的决策树

一旦使用训练数据集创建了决策树，就可以使用它来对新的对象进行分类。为了进行分类，我们从树的根节点开始，沿着与该对象属性值相关联的分支向下，直到到达代表该对象的叶节点为止。

显然，对于一个训练实例集，可能产生很多决策树。基本思想是挑选能够对最多的未知样本正确分类的决策树（这是归纳过程的本质）。一种做法是由训练集产生所有可能的决策树，从中选择最简单的决策树。作为替代，可以用这样的方法构造树，使得结果树是最佳的。在 ID3 (Quinlan 的决策树归纳-3)，他们使用信息理论度量使用特定属性作为节点的“信息增益”，以决定特定节点上的属性。

尽管决策树已经成功地用于大量算法，但是它们也有一些缺点。第一，即使对于小型训练集，决策树也可能相当大，因此难以理解。Quinlan[1]指出，无论决策树的功能多么好，能否用决策树这样的难以理解的结构描述知识仍然是有问题的。第二，当检验数据集中对象的属性具有缺失值时，树的性能可能有问题。此外，树节点中属性的次序可能对性能具有负面影响。

决策树的主要优点是它的执行效率，这主要归结于它的简约表示和执行能力。然而，它们缺乏语义网络和知识表示的其他一阶谓词逻辑方法的语义表达能力。

神经网络：让机器模拟人类的智能行为是人工智能研究者的长期目标。人工智能研究者从多方面获得灵感，如心理学、认知科学和神经计算 (neurocomputing)。

神经网络是稠密的、互联的处理单元组成的网络，通过规则调整单元之间连接强度，以响应外部提供的数据。网络的总体行为由它的连接性，而不是由任何单元的具体操作确定。神经网络的不同拓扑结构适合不同的任务。例如，Hopfield 网络适用于优化问题，多层感知器适用于分类问题，而 Kohonen 网络适用于编码。图 3-8、图 3-9 和图 3-10 显示了这些网络。

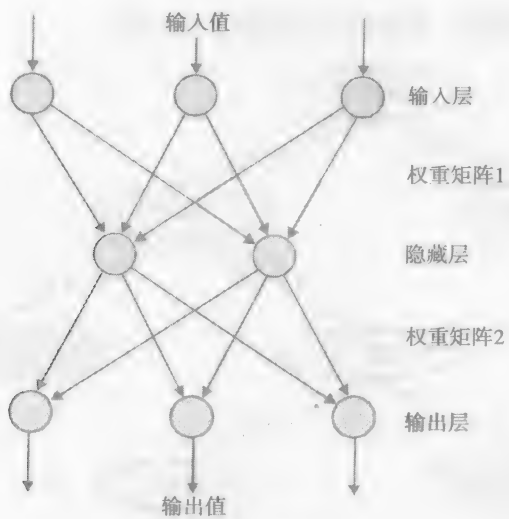


图 3-8 多层感知器

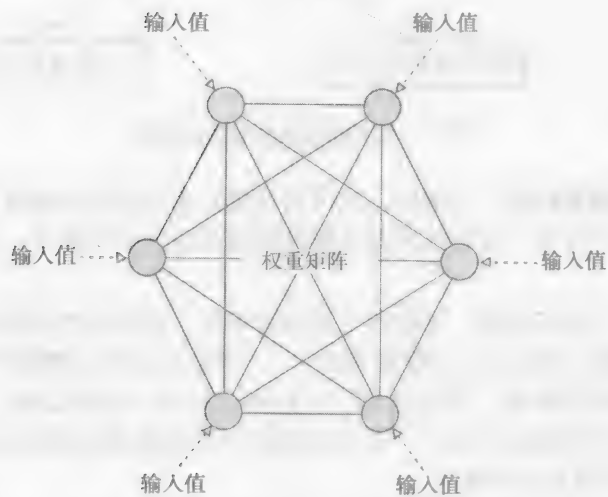


图 3-9 Hopfield 网络

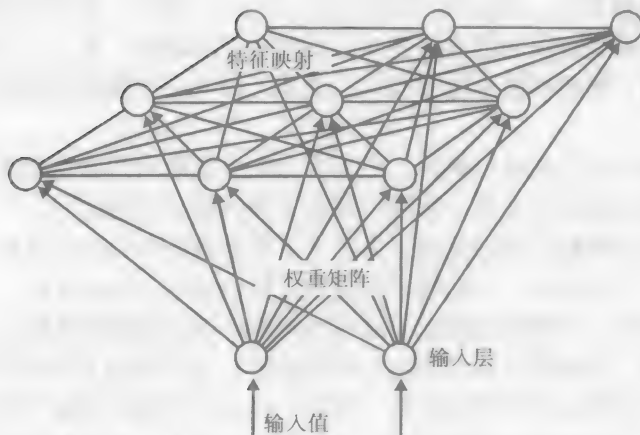


图 3-10 Kohonen 特征映射

尽管神经网络看起来是一个很吸引人的概念，但是它有许多缺点。第一，与其他学习方法相比，学习过程非常慢。对于用户来说，学习得到的知识很难解释(与使用决策树的缺点一样)。很难将用户干预结合到学习过程中，而这对数据挖掘应用来说是需要的。然而，对于现实世界中的噪声数据，神经网络比符号学习技术(基于决策树的规则归纳)的性能好。

规则：规则可能是最常见的知识表达形式。规则是一个条件语句，对特定的条件集指明动作，通常表示为 $X \rightarrow Y$ 。动作 Y 通常称为规则的后件(consequent)，而条件集 X 是规则的前件(antecedent)。规则集是 IF-THEN 语句的无结构组。

对于表 3-3 中的数据，导出的规则是：

```
IF age = "≤30" AND student = "no" THEN buys_computer = "no"
IF age = "≤30" AND student = "yes" THEN buys_computer = "yes"
IF age = "31-40" THEN buys_computer = "yes"
IF age = ">40" AND credit_rating = "excellent" THEN buys_computer
    = "yes"
IF age = "≤30" AND credit_rating = "fair" THEN buys_computer = "no"
```

表 3-3 产生规则的数据

Age	Income	Student	Credit-rating	Buys-computer
≤30	high	no	fair	no
≤30	high	no	excellent	no
31-40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31-40	low	yes	excellent	yes
≤30	medium	no	fair	no
≤30	low	yes	fair	yes
>40	medium	yes	fair	yes
≤30	medium	yes	excellent	yes
31-40	medium	no	excellent	yes
31-40	high	yes	fair	yes
>40	medium	no	excellent	no

规则作为一种知识表示方法流行的主要原因是它的形式简单。它们容易解释，因为不像决策树和神经网络，它们是非常直观和自然的知识表示形式。此外，规则系统是无结构的，不那么严格，这在基于知识的系统开发的早期阶段是一个优点。

但是，用规则表示知识也有许多缺点。规则缺乏变化并且是无结构的，它们的格式不足以表示许多类型的知识，如因果知识。随着系统中规则数量的增加，系统的性能降低，并且系统越来越难维护和修改。不能向系统随意地添加新的规则，因为它们可能与系统中已有的规则矛盾，导致错误的结论。基于规则的系统的性能退化不是适度的。

如果不是不可能的话，基于规则的表示缺乏结构也使得它很难对现实世界建模。因此，希望能够进行部分推理并且随规则数量增加性能缓慢降低的更有组织和结构化的知识表示。

3.5.2 聚类学习算法的输出

在使用聚类算法从实例学习时，输出采用图表形式展示实例如何划分到簇中。

表形式：这里，我们将每个对象与一个簇号相关联。尽管大部分算法只允许每个对象属

于一个簇，但是某些算法以一定的概率而不是无条件地将实例关联到簇。在这种情况下，每个实例有一个属于每个簇的概率或隶属度，如表 3-4 所示。

表 3-4 簇隶属度

	1	2	3		1	2	3
a	0.4	0.1	0.5	e	0.4	0.2	0.4
b	0.1	0.8	0.1	f	0.1	0.4	0.5
c	0.3	0.3	0.4	g	0.7	0.2	0.1
d	0.1	0.1	0.8	h	0.5	0.4	0.1

Venn 图：有些聚类算法允许一个实例以全隶属函数属于多个簇（见图 3-11）。因此，这种图将实例安排在二维平面，并绘制代表每个簇的重叠子集。

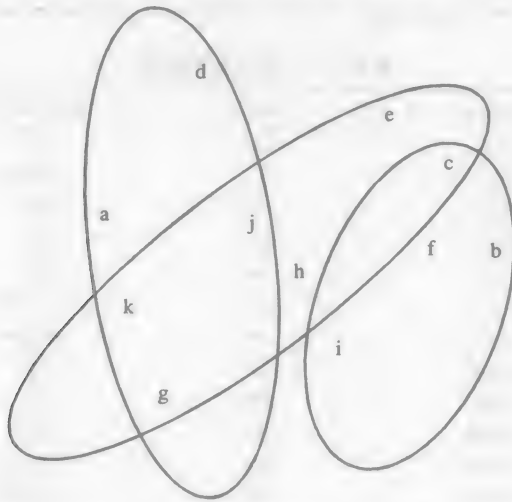


图 3-11 显示簇隶属度的 Venn 图

树状图：许多算法产生图 3-12 所示的簇的层次结构，使得顶层实例空间被划分成几个簇，每个簇又在下一层被划分成子簇，如此下去。这种聚类用树状图表示（dendron 在希腊语中是树的意思）；它基本上是一棵层次结构的树。

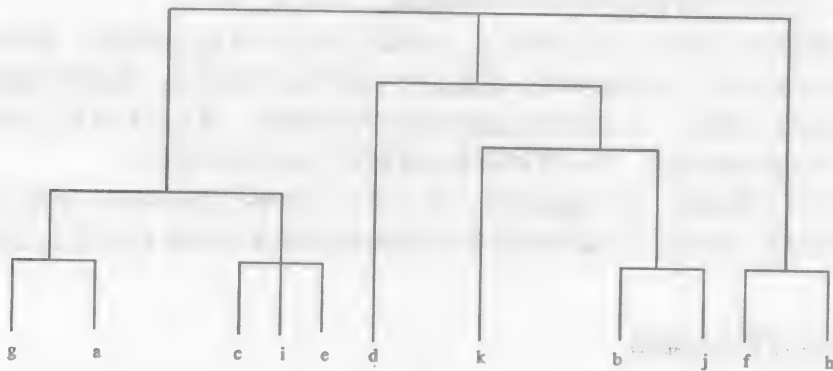


图 3-12 树状图

自组织映射: Kohonen 的自组织映射(SOM)[2]是一种自动安排高维统计数据的工具。该映射试图使用一个受限的模型集合,以最精确的形式表示输入样本。模型在映射栅格中也是有序的,使得相似的模型相互靠近,而不相似的模型相互远离。SOM 在聚类、抽象和通过维归约可视化方面是有用的。SOM 的非监督学习模式使得它适用于不能标记输入数据的应用。图 3-13 显示了关于涉及娱乐主页的聚类的 SOM。



图 3-13 10 000 个涉及娱乐主页的 SOM 映射

生物科学家是聚类算法的主要使用者。伦敦的一位内科医生 John Snow 在地图上绘制了 19 世纪 50 年代霍乱爆发时死亡者的位置(见图 3-14)。结果显示病例聚集在某些存在被污染的水井的十字路口附近——这样就揭示了问题和解决方案。或许, 这是生物科学聚类应用的开始。

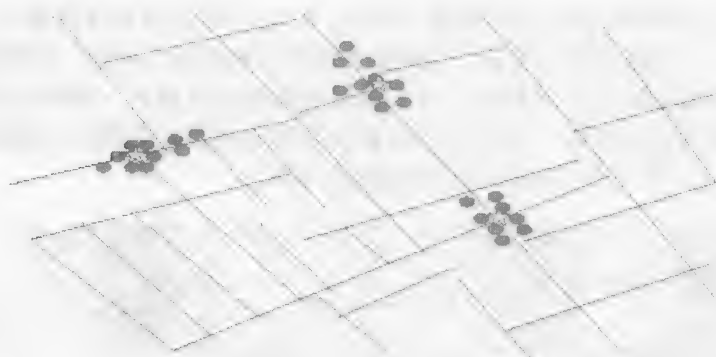


图 3-14 霍乱的位置

3.5.3 关联规则的输出

给定一个事务集，其中事务是项(商品)的集合。关联规则是形如 $X \Rightarrow Y$ 的表达式，其中 X 和 Y 是项集。关联规则的一个例子是“包含啤酒的事务 30% 也包含尿布；所有事务的 2% 都包含这两种商品。”这里，30% 是该规则的置信度，而 2% 是该规则的支持度。问题是找出满足用户定义的最小支持度和最小置信度约束的所有事务规则。表 3-5 显示了一个事务集，而表 3-6 显示了导出的关联规则。

表 3-5 事务数据(人)

RecordID	Age	Married	NumCars
100	23	No	1
200	25	Yes	1
300	29	No	0
400	34	Yes	2
500	38	Yes	2

表 3-6 关联规则(最小支持度 = 40%，最小置信度 = 50%)

规则(样本)	支持度	置信度
(Age: 30 - 39) and (Married: Yes) \geq (Num Cars: 2)	40%	100%
(Num Cars: 0 - 1) \geq (Married: No)	40%	100%

3.5.4 用于数值预测的树的输出

用于数值预测的树基本上是决策树与回归的结合。我们知道，回归是计算预测数值量的表达式的过程。有两种使用回归的树，它们是：

回归树：每个树叶预测一个数值量的“决策树”。预测值是到达该树叶的训练实例的平均值。

模型树：叶节点上具有线性回归模型的“回归树”。这些线性模型近似于非线性的连续函数。

例子：CPU 性能：给出 CPU 性能数据集(cpu.arff)，其中独立变量如下：

- MYCT: 机器周期，单位为纳秒(整型)
- MMIN: 最小主存，单位为千字节(整型)
- MMAX: 最大主存，单位为千字节(整型)
- CACH: 高速缓存，单位为千字节(整型)
- CHMIN: 最小通道，单位为单元(整型)

• CHMAX: 最大通道, 单位为单元(整型)

而 PERF 是性能指数, 被看作独立变量。

PERF: 公布的性能指数(整型)

表 3-7 CPU 性能测量

MYCT	MMIN	MMAX	CACH	CHMIN	CHMAX	PERF
125	256	6000	256	16	128	199
29	8000	32000	32	8	32	253
29	8000	32000	32	8	32	253
29	8000	32000	32	8	32	253
29	8000	16000	32	8	16	132
23	16000	32000	64	16	32	381
23	16000	64000	64	16	32	749
125	2000	8000	0	2	14	41
480	512	8000	32	0	0	47
480	1000	4000	0	0	0	25

上面数据的回归方程是:

$$\text{PERF} = -55.9 + 0.0489\text{MYCT} + 0.0153 * \text{NMIN} + 0.0056 * \text{MMAX} + 0.6410\text{CASH} + (-0.27)\text{CHMIN} + 1.480 * \text{CHMAX}$$

该数据的回归树在图 3-15 中给出。

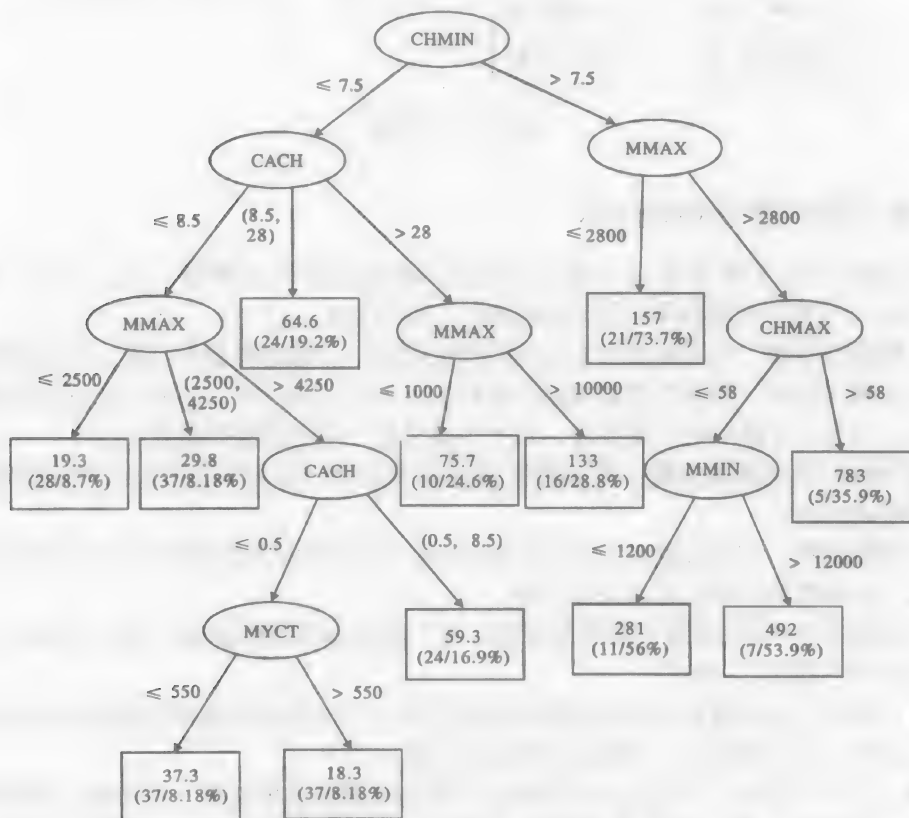


图 3-15 回归树

该数据的模型树在图 3-16 中给出。显示回归方程的表在图的右下角。

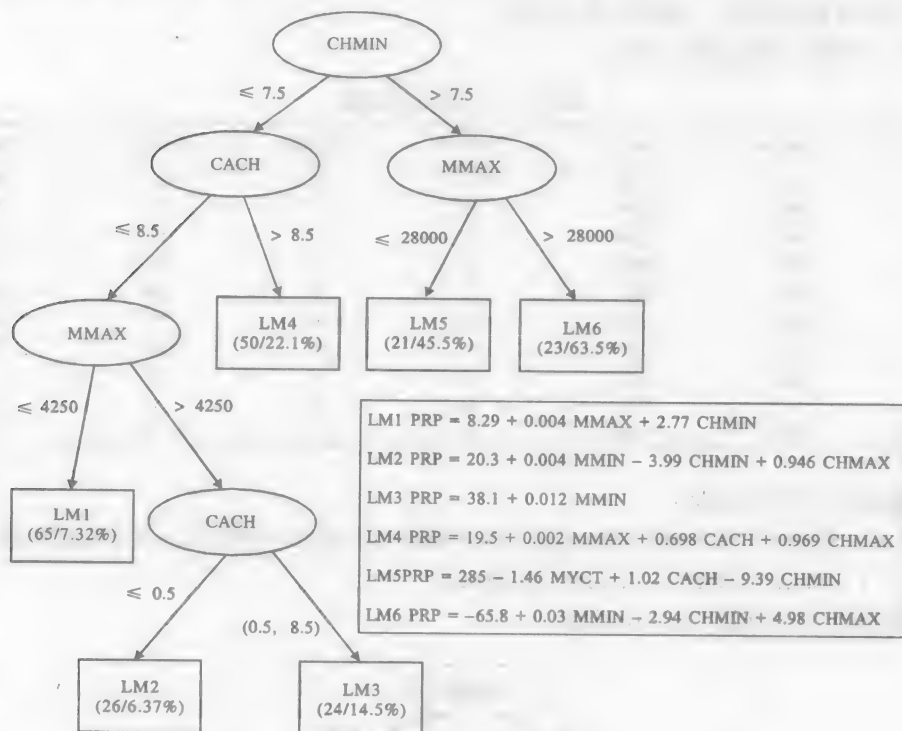


图 3-16 模型树

3.5.5 基于实例的学习和知识表示

基于实例的学习技术通过为每个类保存典型的属性实例而工作。Aha、Kibler 和 Albert[3]定义了基于实例的学习算法具有如下三个特点：

一个相似度函数：告诉算法两个实例的接近程度。尽管这看起来容易，但是选择相似度函数的工作相当复杂，特别是当某些输入是枚举类型时尤其困难。例如，如果我们试图对人进行匹配，并且一个属性是头发颜色，对于头发颜色，距离的意思是什么？

一个“典型”实例选择函数：告诉算法保存哪些实例。如何知道哪些实例是典型的，哪些实例不是典型的？

一个分类函数：该函数就是对给定新的实例，决定如何将它联系到学习过的实例的函数。例如，该函数可以是位置最近的实例。

基于实例的学习 (IBL) 有许多其他名称，有一些基本方法的变种。Aha、Kibler 和 Albert 在他们的文章中提出了三种：

IBL1：存放所有实例并且只找出最邻近的实例——该实例的类就是最邻近的实例的类。然而，这需要存放大量实例，可能需要很大的空间。

IBL2：类似于 IBL1，但是丢弃训练集中已经被正确分类的实例。这节省一些存储空间。

IBL3：像 IBL2 一样，但是对数据做某种假设，并使用统计方法“排除”不相关或噪声实例。

此外，可以使用 k-最近邻 (k-nn) 方法扩展上述方法[4, 5, 6]。我们可以考虑一组最邻

近的点,使用“投票”机制在它们中间选择,而不是只考虑单个最邻近的点并根据它来分类。

基于实例的学习与神经网络具有相同的问题,如系统的“Tweakability[⊖]”——由于必须建立上面所讨论的三个函数,并找出最优的。它们也存在“可提取性(Extractability)”问题,因为它基本上是一种非结构化的学习方法——它只是存放典型值,而不对数据做任何处理。没有任何“概念”以可读的形式产生。

其他问题是:即使对于优化的 IBL3 和 IBL3,有时它们也需要相当大的存储空间。这本身不是问题,但是可能影响最邻近的点的快速搜索。

另一方面,它们容易检验,概念上简单,并且能够以复杂的方式划分空间。

参考文献

- [1] J.R. Quinlan, Induction of decision trees, *Machine Learning* 1, 1986.
- [2] Kohonen, Teuvo, *Self Organizing Maps*, Berlin Heidelberg, New York, 1995.
- [3] David W. Aha, Dennis Kibler, and Marc K. Albert, Instance-based learning algorithms, *Draft Submission to Machine Learning*, 1990.
- [4] T.M. Cover and P.E. Hart, Nearest neighbour pattern classification, *IEEE Transactions on Information Theory*, IT-13(1), pp. 21–27, January 1967.
- [5] Peter E. Hart, The condensed nearest neighbour rule, *IEEE Transactions on Information Theory*, pp. 516–517, May 1968.
- [6] Geoffrey E. Gates, The reduced nearest neighbour rule, *IEEE Transactions on Information Theory*, pp. 431–433, May 1972.

⊖ 译者注: Tweakability 是指无法用简洁的方式来设置对学习算法具有显著影响的一些因子,而需要反复调整。

第4章 决策树——分类和回归树

4.1 引言

分类树是使用树结构算法将数据分成离散类的方法。Breiman[1]在20世纪80年代早期创造了该术语。该技术在医疗、市场调查统计、营销和顾客关系方面得到了很好的应用。例如，一个树结构分类器使用血压、年龄和先前的治疗情况将心脏病患者分成危险和不危险两类。另一种工具可能使用与年龄相关的变量和其他人口统计量决定谁应该出现在邮件发送清单上。预测对直接邮寄广告的反应和确定控制电信业顾客流失的方法都是具体行业的应用。使用分类工具的应用不胜枚举。

决策树的主要作用是揭示数据中的结构化信息。为了解释决策树分类的基本思想，考虑假想的医疗数据。观察表4-1，并不容易看出数据中响应变量“药物”(也称为目标变量或类变量)和解释(预测)变量(性别，年龄，血压)之间的联系，尽管对于小数据集，你或许能够立即从中得到一些结论。如果将该数据集提供给决策树软件，它可能产生一棵图4-1所示的树。

表4-1 医疗数据

0	性别	年龄	血压	药物
1	男	20	正常	A
2	女	73	正常	B
3	男	37	高	A
4	男	33	低	B
5	女	48	高	A
6	男	29	正常	A
7	女	52	正常	B
8	男	42	低	B
9	男	61	正常	B
10	女	30	正常	A
11	女	26	低	B
12	男	54	高	A

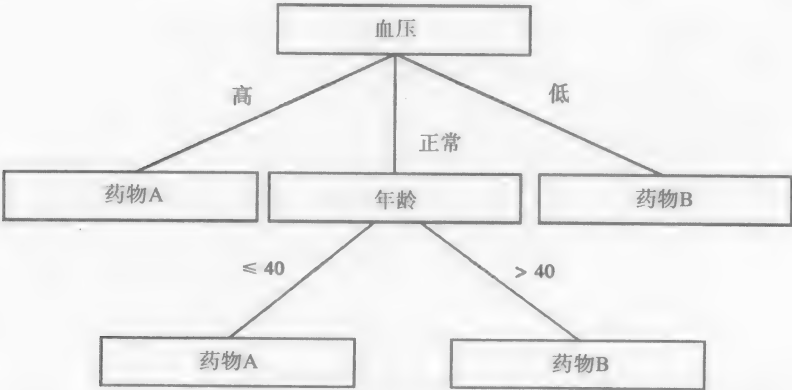


图4-1 医疗数据的决策树

可以看出,该树汇总了数据,并揭示了其中隐藏的结构。由此,我们可以得到如下处方规则:

如果血压高,则采用药物 A。

如果血压低,则采用药物 B。

如果血压正常并且年龄小于或等于 40,则采用药物 A,否则采用药物 B。

在实际开具医疗处方时,内科医生可能使用多个变量作为预测变量,从而准确诊断出疾病。在大部分情况下,他的诊断是基于书籍和杂志上记录的事实和他过去处理类似案例的经验。基于对病人的观察,执业医师可以根据在以往经验中学到的知识使用上面所述的方法转换成规则。如果被大量数据(案例)支持,这些规则将成为进入医疗部门新医生的十分有用的指南。其中一些规则甚至可能成为试图发现这些规则的理论基础的新一代科学家的研究课题。

在前面介绍的这个例子中,没有训练误差;我们产生的规则至少对上述数据都是 100% 正确。然而,对于实际数据,我们多半不能得到具有 100% 准确率和高支持度的规则。支持度是指满足规则的数据点所占的百分比。我们从该例得到的规则和对应的准确率和支持度是:

如果血压高,则采用药物 A(准确率 100%,支持度 3/12)。

如果血压低,则采用药物 B(准确率 100%,支持度 3/12)。

如果血压正常并且年龄小于或等于 40,则采用药物 A(准确率 100%,支持度 3/12)。

如果血压正常并且年龄大于 40,则采用药物 B(准确率 100%,支持度 3/12)。

为了更好地理解“错误率”和“支持度”的概念,考虑图 4-2 所示的假想的树。为了方便稍后进一步解释,假设我们不再进一步分裂节点。根据数据点的类值进行多数表决,左边的节点标记为“A 节点”,右边的节点标记为“B 节点”。这就产生了规则错误率和支持度的概念。

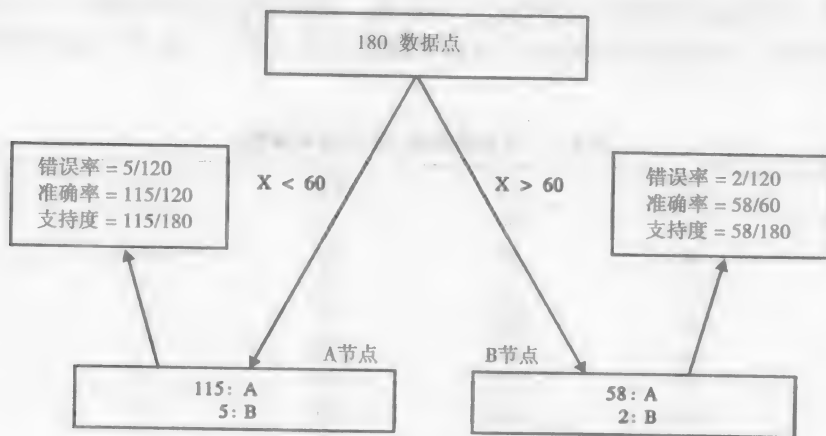


图 4-2 具有不同准确率和支持度的规则

现在,我们考虑决策树(或分类/回归树)算法用于树生长的策略。主要问题是:

- 1) 选择分裂变量的标准。
- 2) 找到被选择的变量的分裂点的标准(连续变量情况)。
- 3) 确定何时停止树生长过程的标准。

在本章中,我们讨论几个算法,它们的主要差别是用来处理上述问题 1 和 2 的标准不

同。如果目标变量(也称为响应变量或类变量)是标称/分类变量(如处方药),则称该树为分类树(classification tree);如果目标变量是连续的(如“收入”),则称该树为回归树(regression tree)。预测变量也可以一般地分为标称的或连续的,并且本章只考虑标称/分类变量。处理连续值变量在第5章讨论,因为大部分实际算法确实在构造树之前先将连续值变量转换成具有离散层次(或区间)的变量。

4.2 构造分类树

4.2.1 用于标称属性的 ID3 算法

ID3 代表归纳决策树(induction decision-tree)版本3,它是一种用来由数据构造决策树的递归过程。我们试探性地选择一个属性放置在根节点,并对该属性的每个值产生一个分支。这样,分裂根节点上的数据集,并移到子女节点,产生一棵局部树(partial tree)。对该划分的质量进行评估。对其他属性重复该过程。每个用于划分的属性产生一棵局部树。然后,根据局部树的质量,选择一棵局部树。这实质上意味选择一个划分属性。换言之,我们根据哪个属性会得到“好的”局部树来选择一个属性。对选定的局部树的每个子女节点重复该过程。这是一个递归过程。如果一个节点上的所有实例都具有相同的类,则停止局部树的生长。

现在,给定一个具有不同类的实例集,我们需要确定使用哪个属性进行划分的标准。考虑表4-2所示的气象数据。由于有4个属性,因此有4棵可能的局部树,在顶层产生的树如图4-3a~图4-3d所示。哪一棵局部树最好?叶节点上显示了“yes”和“no”类的数目。只具有一个类(“yes”或“no”)的叶节点不必再进一步划分,并且到该分支的递归过程将结束。由于我们寻找小树,因此希望停止划分尽可能早地发生。如果我们具有节点纯度的度量,那么应当选择产生最纯子女节点的属性。观察一下图4-3a~图4-3d,并仔细思索你认为哪个属性是最佳选择。

我们需要一种度量来度量节点的纯度,并需要一种度量告诉我们根据一个变量的属性值将一个不纯的节点上的数据划分到其子女后,纯度提高了多少。最为广泛使用的度量是信息熵。

表 4-2 气象数据集(都是标称属性)

序号	天气	气温	湿度	有风	打网球
1	晴	热	高	无	No
2	晴	热	高	有	No
3	多云	热	高	无	Yes
4	雨	温暖	高	无	Yes
5	雨	凉爽	正常	无	Yes
6	雨	凉爽	正常	有	No
7	多云	凉爽	正常	有	Yes
8	晴	温暖	高	无	No
9	晴	凉爽	正常	无	Yes
10	雨	温暖	正常	无	Yes
11	晴	温暖	正常	有	Yes
12	多云	温暖	高	有	Yes
13	多云	热	正常	无	Yes
14	雨	温暖	高	有	No

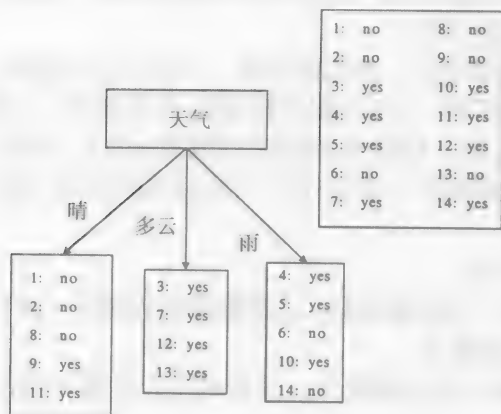


图 4-3a) 基于“天气”划分

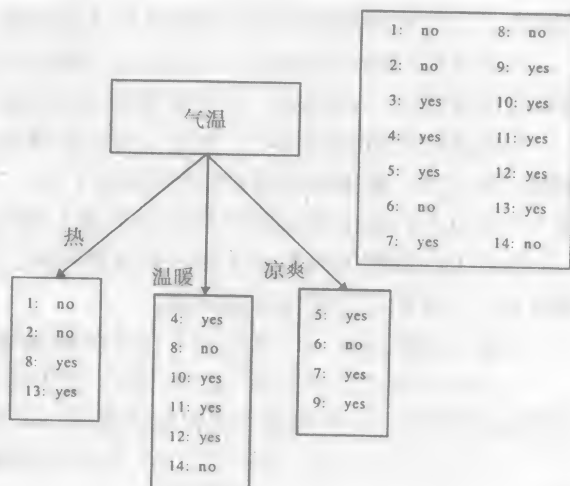


图 4-3b) 基于“气温”划分

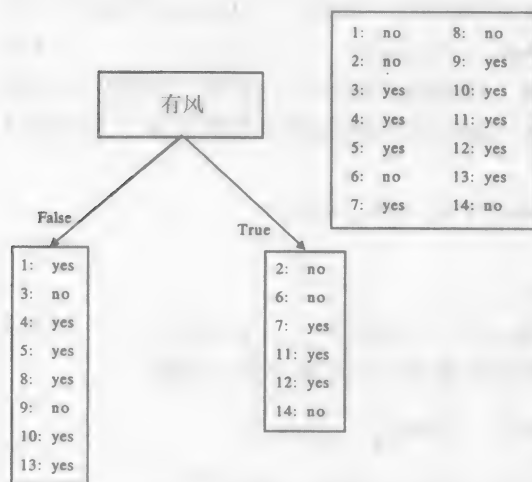


图 4-3c) 基于“有风”划分

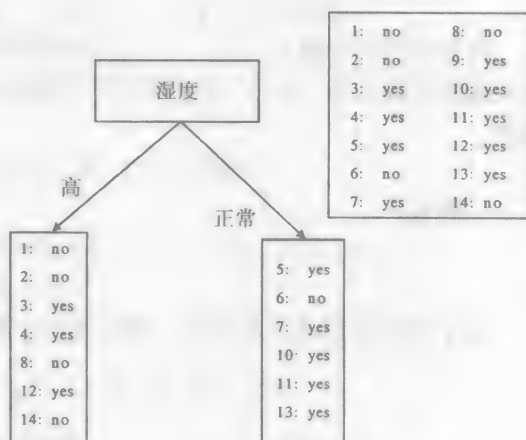


图 4-3d) 基于“湿度”划分

4.2.2 信息论和信息熵

信息论 (information theory) 是数学中的概率论和数理统计的一个分支, 用于处理信息和信息熵、通信系统、数据传输和率失真理论、密码学、信噪比、数据压缩和相关课题。不要将它与图书馆学和信息科学或信息技术相混淆。

Claude Shannon (1916—2001) 被称为信息论之父。他的理论“将信息传输看作一种统计学现象”, 并且为通信工程师提供了一种方法, 使用普通的二进制位流确定通信信道的容量。该理论的信息传输并不“关注信息或消息内容本身”, 尽管与信息论互补的学科关注内容本身, 考虑服从保真标准的消息的有损压缩。信息论的这两个门派联合在一起, 并且通过信息传输定理或源信道分离定理相互印证。信息传输定理证实, 在许多背景下, 使用比特作为信息的通用流通是正确的。

熵 (entropy) 是源于热力学的概念, 但是随后出现信息论中。这两个概念确实具有某些

共同点, 尽管透彻理解这两个领域之后才能发现其中的区别。

热力学熵(thermodynamic entropy) S (通常在化学和热力学中简称熵) 是物理系统中不能用来做功的能量的一种度量。它也是系统无序性的一种度量。

在构造决策树的过程中, 熵定义为无序性度量很合适。我们想选择一个属性划分数据, 使得子女节点上数据的类值(在我们的例子中, “yes”或“no”)大部分都相同(低无序性)。如果一个节点上的数据的类值在可能的类值上均匀分布, 则称节点的熵(无序性)最大。如果一个节点上的数据的类值对于所有数据都相同, 则熵最小。通过分裂, 我们希望得到尽可能纯的节点。这相当于降低系统的熵。

因此, 我们需要一个具有如下性质的信息熵公式:

1) 当一个节点上的“yes”或“no”的个数为零时, 信息熵为零。这意味在构造树时, 该节点仅包含类为“yes”的数据点或仅包含类为“no”的数据点。

2) 当一个节点上的“yes”和“no”的个数相等时, 信息熵最大。这样的节点是最不纯的节点。

此外, 该度量应当可以用于多个类的情况, 而不仅仅是两个类的情况。令人惊奇的是, 结果只有一个函数满足这些性质, 该函数称作信息值或熵。Claude E. Shannon 将其定义为:

$$\text{entropy}(p_1, p_2, \dots, p_n) = -p_1 \log_2 p_1 - p_2 \log_2 p_2 - \dots - p_n \log_2 p_n \quad (4.1)$$

减号是因为分数 p_1, p_2, \dots, p_n 的对数为负, 因此熵实际为正。通常, 对数以 2 为底, 从而熵的单位是位(bit)——计算机中常用的位。熵公式中的参数用分数表示, 其和为 1。例如

$$\text{info}([2, 3, 4]) = \text{entropy}(2/9, 3/9, 4/9)$$

一般地,

$$\begin{aligned} \text{info}([C_1, C_2, \dots, C_n]) &= \text{entropy}(p_1, p_2, \dots, p_n) \\ &= -p_1 \log_2 p_1 - p_2 \log_2 p_2 - \dots - p_n \log_2 p_n \end{aligned} \quad (4.2)$$

由于对数函数的计算方式, 我们可以计算信息度量而不必计算分数。例如,

$$\begin{aligned} \text{info}([2, 3, 4]) &= -\frac{2}{9} \log \frac{2}{9} - \frac{3}{9} \log \frac{3}{9} - \frac{4}{9} \log \frac{4}{9} \\ &= [(-2 \log 2 - 3 \log 3 - 4 \log 4 + 9 \log 9)/9] \end{aligned}$$

这就是实践中计算信息度量的通常办法。让我们看看如何使用信息度量。

4.2.3 构造树

在创建图 4-3 的树结构之前, 训练样本(用来创建树的数据集)在包含 9 个 yes 和 5 个 no 的根节点上, 对应于信息值 $\text{info}([9, 5]) = 0.940$ 位。

在评估图 4-3a 中的第一棵树时, 在叶节点的 yes 和 no 类的个数分别是 $[2, 3]$, $[4, 0]$ 和 $[3, 2]$, 而这些节点的信息值分别是

$$\text{info}([2, 3]) = 0.971 \text{ 位}$$

$$\text{info}([4, 0]) = 0.0 \text{ 位}$$

$$\text{info}([3, 2]) = 0.971 \text{ 位}$$

我们计算它们的平均信息值, 考虑划分到每个分支的实例数。将 5 个实例划分到第一个和第三分支, 4 个实例划分到第二个分支。表 4-3 显示了该划分过程。

表 4-3 根据变量“天气”的值划分

	打网球 = Yes	打网球 = No	合计
晴	2	3	5
多云	4	0	4
雨	3	2	5
合计	9	5	

$$\begin{aligned} \text{info}([2, 3], [4, 0], [3, 2]) &= \frac{5}{14} \left[-\frac{2}{5} \log_2 \left(\frac{3}{5} \right) - \frac{3}{5} \log_2 \left(\frac{3}{5} \right) \right] + \frac{4}{14} \left[-\frac{4}{4} \log_2 \left(\frac{4}{4} \right) - 0 \log_2 0 \right] \\ &\quad + \frac{5}{14} \left[-\frac{3}{5} \log_2 \left(\frac{3}{5} \right) - \frac{2}{5} \log_2 \left(\frac{2}{5} \right) \right] = 0.693 \text{ 位} \end{aligned}$$

这样，图 4-3a 的树导致的信息增益为

$$\begin{aligned} \text{gain}(\text{天气}) &= \text{info}([9, 5]) - \text{info}([2, 3], [4, 0], [3, 2]) \\ &= 0.940 - 0.693 = 0.247 \text{ 位} \end{aligned}$$

它可以解释为在“天气”(outlook)属性上创建分支的信息值。

前进的路是清楚的。我们为每个属性计算信息增益，并选择获得最大信息增益的属性进行划分。

$$\text{gain}(\text{天气}) = 0.247 \text{ 位}$$

$$\text{gain}(\text{气温}) = 0.029 \text{ 位}$$

$$\text{gain}(\text{湿度}) = 0.152 \text{ 位}$$

$$\text{gain}(\text{有风}) = 0.048 \text{ 位}$$

这样，我们选择天气作为树的根节点的划分属性。希望这与你的直观最佳选择一致。这是唯一的选择，其中一个子女节点是最纯的，并且这使它明显优于其他属性。湿度是次佳选择，它产生了一个几乎完全纯的较大的子女节点。

然后，我们递归地继续选择。图 4-4a ~ 图 4-4c 显示了当天气为晴时所到达的节点上的可能的深一层的分支。显然，在天气上的进一步划分不会产生新的结果，因此只考虑其他 3 个属性。每个属性产生的信息增益分别为

$$\text{gain}(\text{气温}) = 0.571 \text{ 位}$$

$$\text{gain}(\text{湿度}) = 0.971 \text{ 位}$$

$$\text{gain}(\text{有风}) = 0.020 \text{ 位}$$

因此，此处我们选择湿度作为划分属性。不再需要对这些节点做进一步划分。

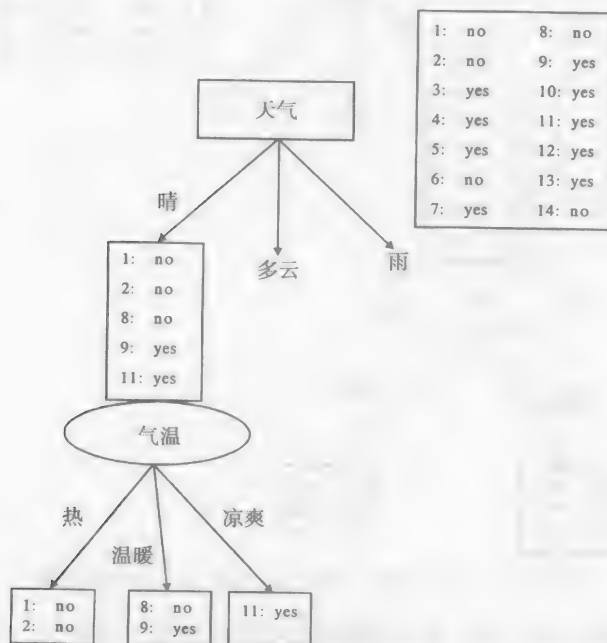


图 4-4a) 天气 = 晴，气温 = {热，温暖，凉爽} 的局部树

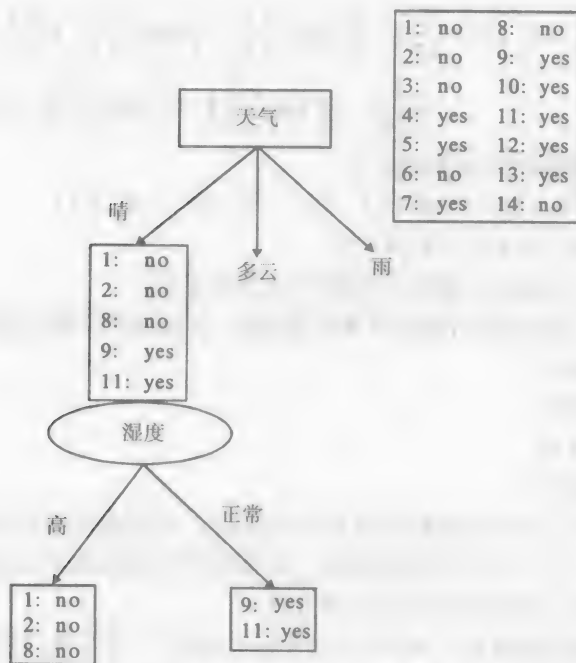


图 4-4b) 天气 = 晴, 湿度 = {高, 正常} 的局部树

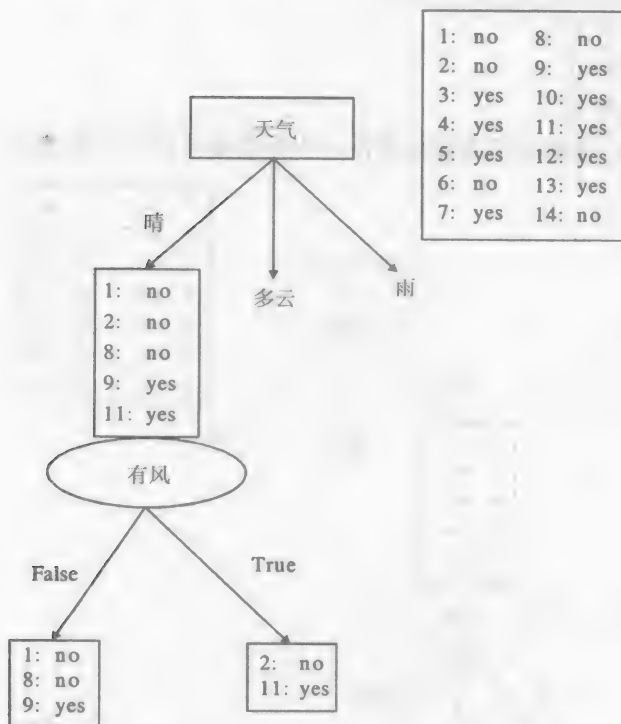


图 4-4c) 天气 = 晴, 有风 = {True, False} 的局部树

现在, 我们对按照“天气 = 雨”分支得到的数据进行划分。图 4-5a ~ 图 4-5c 显示了这些局部树。当节点上的数据使用变量“有风”划分时, 子女节点是纯的。这产生了图 4-6 的气象数据

的决策树。理想情况下,当所有叶节点都是纯的而使过程终止时,即当它们包含的实例都具有相同类时该过程终止。然而,可能无法到达这种结果,因为无法避免训练集包含两个具有相同属性集,但具有不同类的实例。因此,当数据不能进一步划分时,我们停止划分过程。

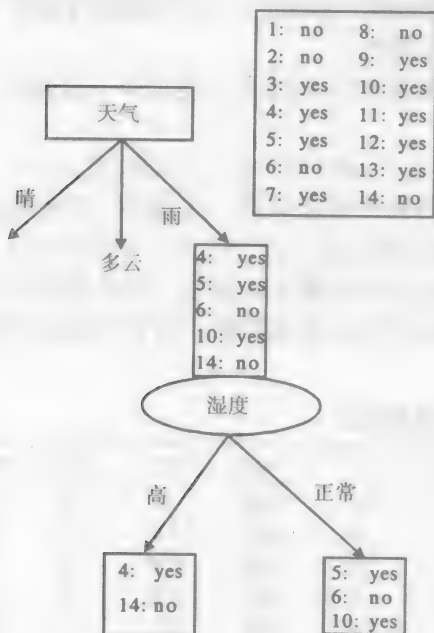


图 4-5a) 天气 = 雨, 湿度 = {高, 正常} 的局部树

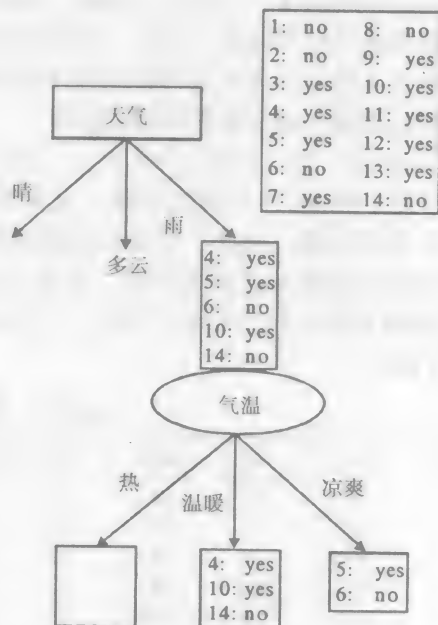


图 4-5b) 天气 = 雨, 气温 = {热, 温暖, 凉爽} 的局部树

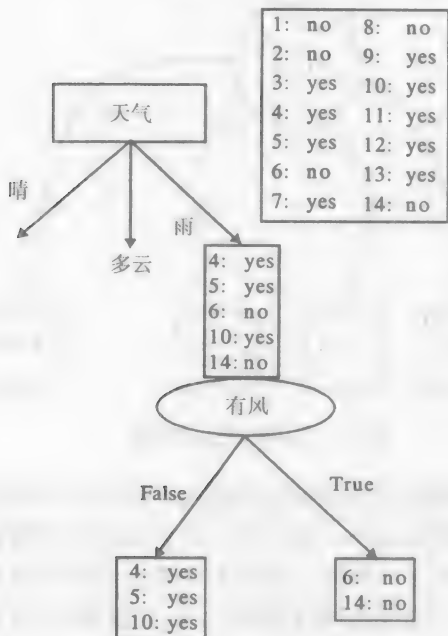


图 4-5c) 天气 = 雨, 有风 = {True, False} 的局部树

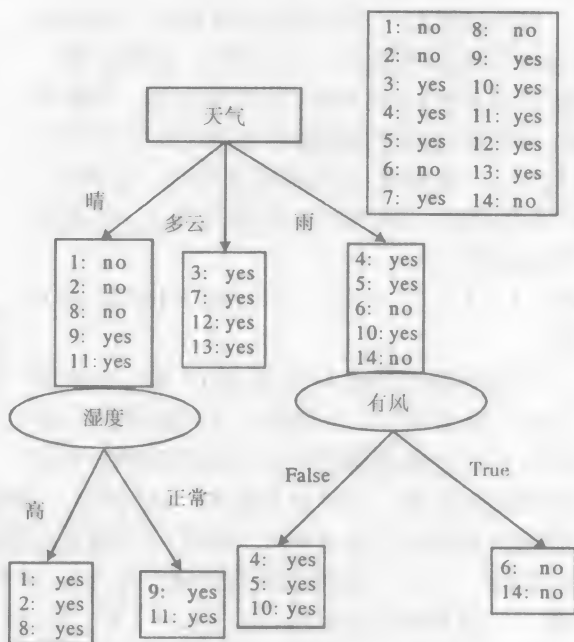


图 4-6 最终的决策树

4.2.4 高分支属性

当某些属性具有大量可能值时,会导致具有许多子女节点的多路分支出现,信息增益的计算就会出现問題。对于这一问题,最好用一个极端的例子来体会:对于数据集中的每个实例,属性都具有不同值。例如,标识码(identification)属性。

表4-4给出了具有这种附加属性的气象数据。在ID码上的分支产生图4-7的树桩。给定该属性的值,确定类需要的信息为

$$\text{info}([0, 1]) + \text{info}([0, 1]) + \text{info}([0, 1]) + \cdots + \text{info}([0, 1]) + \text{info}([0, 1])$$

它等于零,因为14个项均为零。这并不奇怪:ID码属性确定实例,这就无二义地确定了表4-4显示的类。因此,该属性的信息增益恰为根上的信息 $\text{info}([9, 5]) = 0.940$ 位。它大于其他任何属性的信息增益,因而ID码必然被选作划分属性。但是,在标识码上分支对于预测未知实例的类毫无用处,并且对于决策的结构也毫无帮助,而它们都是机器学习的目标。

表4-4 具有标识码的气象数据

ID 码	天气	气温	湿度	有风	打网球	ID 码	天气	气温	湿度	有风	打网球
a	晴	热	高	无	No	h	晴	温暖	高	无	No
b	晴	热	高	有	No	i	晴	凉爽	正常	无	Yes
c	多云	热	高	无	Yes	j	雨	温暖	正常	无	Yes
d	雨	温暖	高	无	Yes	k	晴	温暖	正常	有	Yes
e	雨	凉爽	正常	无	Yes	l	多云	温暖	高	有	Yes
f	雨	凉爽	正常	有	No	m	多云	热	正常	无	Yes
g	多云	凉爽	正常	有	Yes	n	雨	温暖	高	有	No

总体效果是:信息增益度量趋向于选择具有大量可能值的属性。作为补偿,通常使用一种称作增益率(gain ratio)的度量变型。增益率通过考虑属性划分数据集产生的子女节点的个数和大小,忽略关于类的信息导出。在图4-7所示的情况中,所有的计数值均为1,因此划分信息值是

$$\text{info}([1, 1, \cdots, 1]) = -1/14 \times \log 1/14 \times 1/14$$

因为相同的分数 $1/14$ 出现14次。上式等于 $\log 14$ 或 3.807 位,这是一个很高的值。因为划分的信息值是确定每个实例指派到哪个分支所需要的位数,并且分支越多,该值越大。增益率通过用原来的信息增益(本例为 0.940)除以属性的该信息值 3.807,得到ID码属性的增益率值 0.246。回到表4-3所示的气象数据的局部树,“天气”将数据集划分成大小为5、4和5的三个子集,因此无须关心子集中涉及的类,它具有固有信息值 $\text{info}([5, 4, 5]) = 1.577$ 。正如我们所看到的,对于假想的ID码这样具有较高分支的属性,该固有信息值较高。再重复一次,信息增益除以该固有信息值得到增益率。

图4-3a~图4-3d的树桩的计算结果汇总在表4-5中。

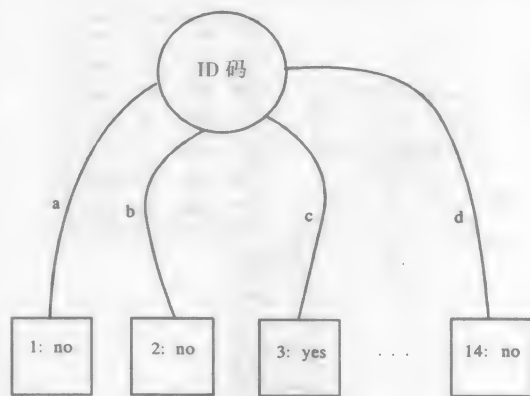


图4-7 ID码属性的树桩

表 4-5 决策变量的增益率

天气	气温
Info = 0.693	Info = 0.911
Gain = 0.940 - 0.693 = 0.247	Gain = 0.940 - 0.911 = 0.029
Split info = info([5, 4, 5]) = 1.577	Split info = info([4, 6, 4]) = 1.362
Gain ratio = 0.247/1.577 = 0.156	Gain ratio = 0.029/1.362 = 0.021
湿度	有风
Info = 0.788	Info = 0.892
Gain = 0.940 - 0.788 = 0.152	Gain = 0.940 - 0.892 = 0.048
Split info = info([7, 7]) = 1	Split info = info([8, 6]) = 0.985
Gain ratio = 0.152/1 = 0.152	Gain ratio = 0.029/1.362 = 0.049

使用“天气”的效果仍然最好，但是现在“湿度”成了有力的竞争者，因为它将数据划分成两个子集，而不是三个。在这个例子中，假想的 ID 码属性的信息增益率为 0.246，仍然好于这 4 个属性中的任何一个。然而，它的优势已经大大降低。在实际实现中，使用特殊的检验以确保不在这种无用的属性上划分。

遗憾的是，在某些情况下，增益率补偿过分，并且可能导致优先选择这样的属性，仅仅因为它的固有信息比其他属性低得多。标准的做法是选择最大化增益率的属性，只要该属性的信息增益至少与所考察的所有属性的平均信息增益一样大。

4.2.5 从 ID3 到 C4.5

决策树归纳(有时也称决策树的自顶向下归纳)的分治技术由澳大利亚悉尼大学的 Ross Quinlan[2]开发并经过多年优化。尽管其他人也在研究类似的方法，但是 Quinlan 的研究总是站在决策树归纳的最前沿。所介绍的使用信息增益标准的方案基本上与 ID3 相同。增益率的使用正是多年来对 ID3 的诸多改进之一；Quinlan 形容它在众多环境下具有鲁棒性。尽管这是一个实际的解决方案，但是它牺牲了信息增益标准的某些优雅和整洁的理论动机。对 ID3 的一系列改进在称作 C4.5 的有影响的、广泛使用的决策树归纳系统中达到高峰。这些改进包括处理数值属性、缺失值、噪声数据和由决策树产生规则的方法。

4.2.6 形象化地理解 ID3 和 C4.5 算法

例 4.1 考虑表 4-6 所示的关于昆虫类数据的部分列表。

表 4-6 昆虫数据

昆虫 ID	腹部长度	触角长度	昆虫类	昆虫 ID	腹部长度	触角长度	昆虫类
1	2.7	5.5	蝗虫	6	2.9	1.9	蝗虫
2	8.0	9.1	蛴螬	7	6.1	6.6	蛴螬
3	0.9	4.7	蝗虫	8	0.5	1.0	蝗虫
4	1.1	3.1	蝗虫	9	8.3	6.6	蛴螬
5	5.4	8.5	蛴螬	10	8.1	4.7	蛴螬

图 4-8 给出了对应的特征空间和决策树。基于“腹部长度”的垂直切割给我们一个仅包含蛴螬的节点(图的右节点)。左节点是一个不纯的节点。现在，基于变量“触角长度”在左部节点数据上的水平切割产生最终的决策树，其所有叶节点都是纯的节点。我们称这种类型的切割为轴平行切割。大部分决策树算法只做轴平行切割。

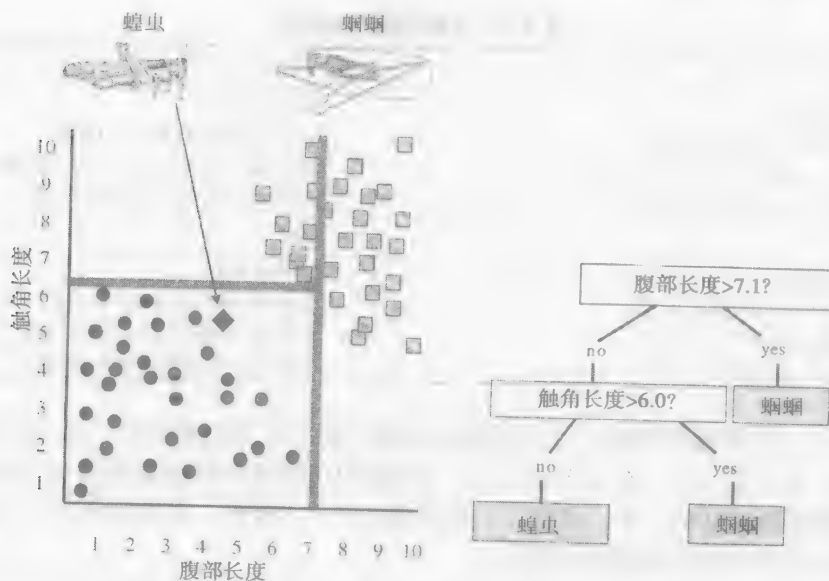


图 4-8 昆虫数据的特征空间和决策树

例 4.2 图 4-9 显示了男性和女性(人)的某些特征的数据。图 4-10 显示了对应的决策树。


图片	姓名	头发长度	体重(公斤)	年龄	类
	Soman	0"	60	36	男
	Sandhya	10"	38	34	女
	Akshay	2"	22	10	男
	Shivani	6"	21	8	女
	Nimmy	4"	10	5	女
	Narayanan	1"	57	70	男
	Sai Lakshmi	8"	40	41	女

图 4-9 女性和男性分类数据



图片	姓名	头发长度	体重(公斤)	年龄	类
	Ram Singh	10"	47	38	男
	Nagabhushan	6"	53	45	男

图 4-9 (续)

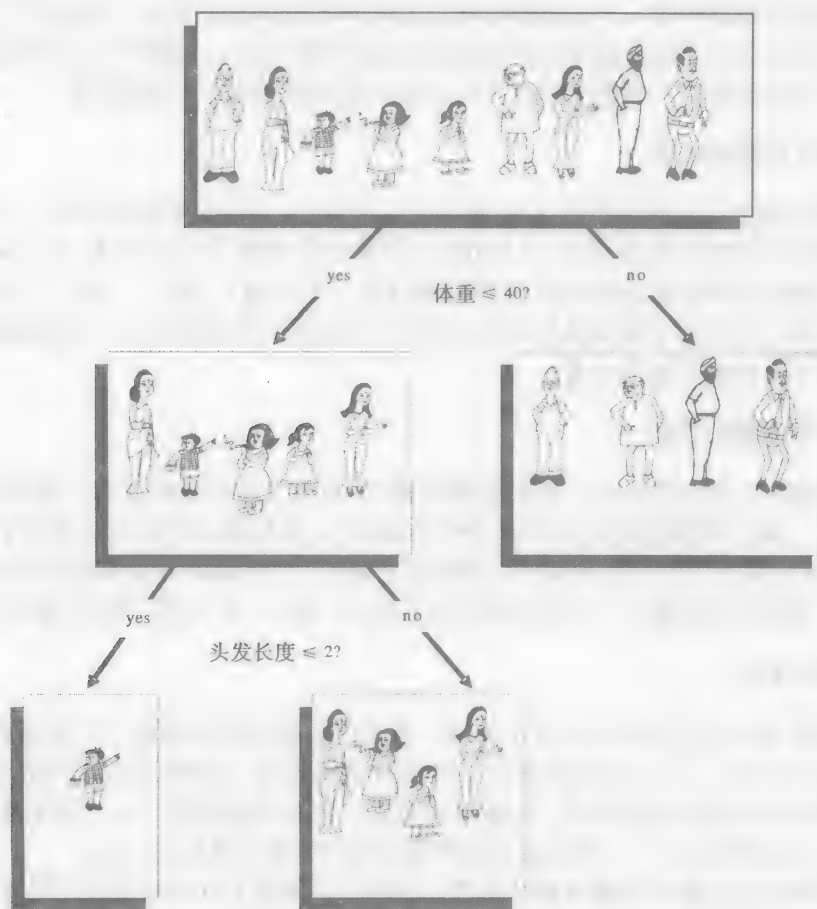


图 4-10 女性和男性分类的决策树

4.3 CHAID

CHAID (Chi-square Automatic Interaction Detection, 卡方自动交互检测) 首次出现在 G. V. Kass 博士发表于 1980 年“应用统计学”杂志上的一篇题为“An Exploratory Technique for Investigating Large Quantities of Categorical Data”的文章中[3]。该过程是一种称作自动交互检测 (Automatic Interaction Detection, AID)[4] 的早期技术的衍生物, 使用卡方统计量作为主要工具。(建议不熟悉卡方统计量的读者阅读本章最后的参考文献。)

像 ID3 一样, CHAID 关注基于一些其他变量(称作预测变量)预测单个变量(称作依赖变量)。

引述“应用统计学”中的文章[3], “CHAID 将数据划分成互斥的、穷举的、能够恰当描述依赖变量的子集”。像 ID3 一样, CHAID 是一种迭代技术, 它单独考察预测子(探查变量)并按照它们的统计显著性指示的次序利用它们。CHAID 分析是非常简单和直观的。CHAID 首先确定哪个预测变量能最有效地区分总体中依赖变量的不同级别(level)。然后, 它根据该变量的重要类别(级别)划分总体。例如, 在汽车保险公司, 问题是根据分类变量预测意外事故发生的频率或纯保险费。如果分析发现年行车里程是区分险损赔偿频率的最佳变量, 则使用重要的行车里程级别(如 2500 以下, 2501 ~ 5000 等)划分总体。然后单独地考察每个划分, 确定剩余的变量中哪一个变量能最有效地区分该划分中的风险。继续进行该过程, 直到考察完所有变量为止。那些重要的变量触发数据的再次划分, 而那些对于该划分不重要的变量被丢弃。结果类似于一棵倒置的树, 每个分支标识总体的一个重要子群。

4.3.1 CHAID 的数学工具

CHAID 涉及基本卡方相依检验(Chi-Square Contingency Test)的复杂应用, 卡方相依检验在每本基础统计学教程中都有介绍。CHAID 以两种方式使用卡方统计量(Chi-Square statistic)。首先, 它确定预测变量的级别(不同的属性值, 如气温 = “热”, 气温 = “凉爽”等)能否合并。一旦所有的预测子级别都被压缩(或合并)成最小的显著形式, 它就确定哪个预测子在区分依赖变量级别方面最显著。

4.3.2 CHAID 变量的类型

在更详细地介绍该过程之前, 我们必须解释 CHAID 支持的数据类型。依赖变量必须划分成离散类别。假定预测变量可以具有三种不同形式: 具有隐含序的单调预测子(如驾驶员的年龄), 不具有隐含序的自由预测子(如地域)和除一个级别外都遵循特定序的浮动预测子。在与其他单调变量合取中, 浮动预测子允许使用“缺失”或“未知”级别(称为浮动级别)。

4.3.3 CHAID 算法

假定依赖变量(目标变量)具有 d 个级别, 并且所分析的具体预测子变量具有 c 个级别。这些数据可以汇总在一个 $c \times d$ 相依表中。CHAID 分析的第一步的目标是压缩该 $c \times d$ 表的行, 使得它仅包含显著不同的级别。用数学术语说, 我们希望将该 $c \times d$ 表归约成最重要的 $j \times d$ 表, 其中 j 的范围为 $2 \sim c$ 。我们选择具有最显著卡方统计量的 $j \times d$ 表。

允许的级别合并依赖于预测变量的类型。尽管自由预测子可以以任意方式合并, 但是对于单调变量而言, 只能合并相邻的级别。浮动预测子级别的合并也限于相邻级别, 唯一的例外是浮动类别, 它可以单独也可以与任何其他组群合并。

最佳 $j \times d$ 表的实际计算一般需要进行动态规划, 以考察所有可能的排列。对于单调预测子, 该计算的量级是 c^2 。对于自由预测子, 解的量级是 2^c (注意, c 是预测变量的级别数)。显然, 对于考察大量预测子变量, 这样的计算量使得动态规划成为一种不切实际的方法。因此, Kass 博士[3]开发了一种替代的方法, 它类似于逐步和分段回归中使用的技术。它不能保证得到最优解, 但是可以产生实践中相当满意的结果。

一旦预测子级别合并之后, 算法必须确定压缩后的相依表的显著性。如果表没有被归约, 则显著性是概率计算值的补, 假定自由度为 $(j-1)(d-1)$, 它可以通过查任意卡方表确定。

然而, 如果相依表被归约, 则算法确保结果表的大小是最好的。因此, 显著性必须反映

这一事实：该表不能被孤立地考虑，而应当在所有可能的 $j \times d$ 表背景下考虑。因此，仅仅确定与计算的卡方统计量相关联的显著性是不够的。Kass 博士[3]使用涉及同时考虑所有 $j \times d$ 表的显著性。使用 Bonferroni 的概率定理，他计算所有 $j \times d$ 表的显著性的一个下界。通过将未归约的表的显著性乘以一个称作 Bonferroni 乘子 (Bonferroni multiplier) 的因子，确定该显著水平。

该乘子对应于可以将 c 个级别归约到 j 个组群的方法数，并且因预测变量的类型而异。对于单调变量，因为只能合并相邻级别，所以 Bonferroni 乘子为：

$$\binom{c-1}{j-1}^{\ominus}$$

自由预测子可以以任意方式组合，其 Bonferroni 乘子为：

$$\sum_{i=0}^{j-1} (-1)^i \frac{(j-i)^c}{i!(j-i)!}$$

毫不奇怪，浮动预测子的 Bonferroni 乘子最复杂，它的形式如下：

$$\binom{c-2}{j-2} + j \binom{c-2}{j-1}$$

我们用一个数值例子来解释乘子的使用。一个具有检验统计量值 23.3 的 4×5 的未归约的相依表具有显著性 0.025 (假设自由度为 12)。然而，如果从 6×5 的相依表开始，并将它降维到 4×5 ，则对于单调预测变量，相同的检验统计量将反映 0.1 ($= 4 \times 0.025$) 的显著性。如果使用 5% 的显著水平作为临界点，则未归约的表将被认为是显著的，而归约表则不是显著的。

关于 Bonferroni 乘子的更全面的解释，建议读者阅读文献[3]。

每次迭代的最后一步是确定具有最高显著性的预测变量对于数据划分而言是否足够显著。如果是，则将数据基于该变量的重要级别划分。如果不是，则过程停止。

4.3.4 CHAID 算法描述

下面是 CHAID 算法的描述，它在 Kass 博士 1980 年的文章[3]中给出。

步骤 1 对于每个预测变量 X ，找出 X 的类别对，它们关于目标变量差别最不显著 (即最大的 p 值)。使用一个以 X 的类别为行，目标变量的类别为列的二路交叉表找出 p 。使用卡方检验来检验显著性。

步骤 2 对于 X 的具有最大 p 值的类别对，将 p 值与预先指定的 α_{merge} 比较。

如果 p 值大于 α_{merge} ，则将该类别对合并成一个复合类别。结果形成 X 的新的类别集，并且从步骤 1 开始该过程。

如果 p 值小于 α_{merge} ，则转到步骤 3。

步骤 3 使用适当的 Bonferroni 调整，为 X 的类别集和目标变量的类别集计算调整后的 p 值。

步骤 4 选择这样的预测变量 X ：它具有最小的、调整后的 p 值 (最重要的一个预测子)。将它的 p 值与预先指定的 α_{split} 比较。

如果该 p 值小于或等于 α_{split} ，则按照 X 的类别集分裂该节点。

如果该 p 值大于 α_{split} ，则不分裂该节点。该节点是一个终端节点。

⊖ 组合公式采用国内流行的记号，下同。——译者注

步骤5 继续树生长过程，直到满足终止条件为止。

4.3.5 将 CHAID 用于气象数据

现在，我们通过将 CHAID 算法用于一个简单数据集来介绍它的步骤。该数据集(参见表 4-2)只包含一个分类目标变量打网球(Y)和 4 个探测变量天气(X_1)、气温(X_2)、湿度(X_3)和有风(X_4)。变量 Y 有两个级别， $Y = \{\text{yes}, \text{no}\}$ 。变量 X_1 有 3 个级别， $X_1 = \{\text{晴}, \text{多云}, \text{雨}\}$ 。变量 X_2 有 3 个级别， $X_2 = \{\text{热}, \text{温暖}, \text{凉爽}\}$ 。变量 X_3 有两个级别， $X_3 = \{\text{高}, \text{正常}\}$ 。变量 X_4 有两个级别， $X_4 = \{\text{TRUE}, \text{FALSE}\}$ 。

CHAID 算法中的步骤是：

1) 计算响应变量打网球在根节点的分布，如表 4-7 所示。

2) 对于每个预测变量 X ，找出它的关于 Y 在该节点的分布差别最不显著的类别对(即具有最大 p 值)。用于计算 p 值的方法依赖于 Y 的度量级别。在这个例子中， Y 是分类的，因此需要进行一系列卡方检验。

在该节点中，探测变量天气(X_1)和目标变量打网球之间的联系由下面的交叉表(见表 4-8)给出：

表 4-7 在根节点分裂

类别	实例数	%
Play = yes	9	64.29
Play = no	5	35.71

表 4-8 基于变量“天气”分裂

天气(X_1)	打网球 = yes	打网球 = no	行合计
$X_1 = \text{晴}$	2	3	5
$X_1 = \text{多云}$	4	0	4
$X_1 = \text{雨}$	3	2	5
列合计	9	5	14

由于 $X_1 = \text{天气}$ 有 3 个类别，因此有 $C_3^2 = 3$ 个 2×2 的子交叉表需要考虑。它们如表 4-9a ~ 表 4-9c 所示。

表 4-9 合并变量“天气”级别的显著性检验

a)	天气(X_1)	打网球 = yes	打网球 = no
	$X_1 = \text{晴}$	2	3
	$X_1 = \text{多云}$	4	0
卡方 = 3.6, d.f = 1, p 值 = 0.05778			
b)	天气(X_1)	打网球 = yes	打网球 = no
	$X_1 = \text{晴}$	2	3
	$X_1 = \text{雨}$	3	2
卡方 = 0.4, d.f = 1, p 值 = 0.5271			
c)	天气(X_1)	打网球 = yes	打网球 = no
	$X_1 = \text{雨}$	3	2
	$X_1 = \text{多云}$	4	0
卡方 = 2.857, d.f = 1, p 值 = 0.0909			

然后，算法识别具有最大 p 值的 X_1 (天气)的类别对，并与预先指定的 α_{merge} 比较(默认值为 0.05)。在这个例子中，由于“天气 = 晴”和“天气 = 雨”对的 p 值 = 0.5271，因此我们合并这些类别，并再次计算交叉表，得到下面的表 4-10。

表 4-10 基于两个级别合并并在变量“天气”上的分裂

天气(X_1)	打网球 = yes	打网球 = no
X_1 = 晴 or 雨	5	5
X_1 = 多云	4	0

卡方 = 3.111, d.f = 1, p 值 = 0.0777

对于变量 X_1 , 我们只有一个这样的表。

现在, 算法对合并后 X_1 的类别和 Y 的类别集使用 Bonferroni 调整计算调整的 p 值。卡方 p 值使用 Bonferroni 乘子调整。由于 X_1 是标称的, 因此其 Bonferroni 乘子用下式计算:

$$B_{\text{free}} = \sum_{i=0}^{r-1} (-1)^i \frac{(r-i)^c}{r!(r-i)!}$$

其中 c 是 X_1 的原来的类别数(3), r = 合并的类别数(2)。这导致调整后的 p 值为 0.3786(= 0.2524×1.5)。

现在, 我们切换到下一个变量“气温”。

3) 用 X_2 (气温)取代 X_1 , 重复对 X_1 的执行步骤。表 4-11 显示了分裂和显著性计算。

表 4-11 合并变量“气温”级别的显著性检验

a)	气温(X_2)	打网球 = yes	打网球 = no
	X_2 = 热	2	2
	X_2 = 温暖	4	2
卡方 = 0.2777, d.f = 1, p 值 = 0.5981			
b)	气温(X_2)	打网球 = yes	打网球 = no
	X_2 = 热	2	2
	X_2 = 凉爽	3	1
卡方 = 0.5333, d.f = 1, p 值 = 0.4652			
c)	气温(X_2)	打网球 = yes	打网球 = no
	X_2 = 温暖	4	2
	X_2 = 凉爽	3	1
卡方 = 0.0783, d.f = 1, p 值 = 0.7781			

最高 p 值是“气温 = 温暖”和“气温 = 凉爽”对的 p 值。因此, 我们合并这些类别, 形成表 4-12 所示的新交叉表。

表 4-12 基于两个级别合并的变量“气温”的分裂

气温(X_2)	打网球 = yes	打网球 = no
X_2 = 温暖 or 凉爽	7	3
X_2 = 热	2	2

卡方 = 0.4977, d.f = 1, p 值 = 0.4804

Bonferroni 调整后, p 值为 1。

接下来, 我们考虑 X_3 = 湿度。这里, 由于类别数为 2, 因此我们只需要做一个交叉表。不需要合并类别。表 4-13 显示了显著性检验。

表 4-13 变量“湿度”的显著性检验

气温(X_3)	打网球 = yes	打网球 = no
X_3 = 高	3	4
X_3 = 正常	6	1
卡方 = 2.8, d.f = 1, p 值 = 0.0942		

最后要考虑的是变量“有风”。交叉表如下面的表 4-14 所示。

表 4-14 变量“有风”的显著性检验

天气(X_4)	打网球 = yes	打网球 = no
X_4 = False	6	2
X_4 = True	3	3
卡方 = 0.9333, d.f = 1, p 值 = 0.3339		

所得到的最小 p 值是变量“湿度”的 p 值, 将“高”划分在一边, “正常”在另一边。这导致树的根节点数据根据变量“湿度”划分。注意, 4.2 节基于熵的划分导致树的根节点基于变量“天气”划分。

4.3.6 单调变量的预测子级别合并

在对单调变量进行 CHAID 分析时, 只试图对连续的级别进行合并。让我们考虑一个例子, 其中要分析的预测子变量是驾驶员的“年龄组”, 而目标变量是在过去的 6 个月中, 购买保险的驾驶员集合的“保险索赔次数”。表 4-15a、表 4-15b 和表 4-15c 分别显示合并前、一次合并和两次级别合并后显著性检验。

表 4-15 一个单调预测子变量的 CHAID 分析例子

a) 合并前					
		依赖变量			
驾驶员年龄	0	1	2	3	合计
20 以下	350	75	50	25	500*
21 ~ 24	584	112	80	24	800*
25 ~ 29	560	84	42	14	700
30 ~ 49	3440	340	140	80	4000
50 ~ 65	2195	180	75	50	2500
65 以上	1245	180	60	15	1500
合计	8374	971	447	208	10000
b) 1 次合并后					
驾驶员年龄	0	1	2	3	合计
24 以下	934	187	130	49	1300*
25 ~ 29	560	84	42	14	700*
30 ~ 49	3440	340	140	80	4000
50 ~ 65	2195	180	75	50	2500
65 以上	1245	180	60	15	1500
合计	8374	971	447	208	10000
c) 2 次合并后					
驾驶员年龄	0	1	2	3	合计
24 以下	934	187	130	49	1300
25 ~ 29	560	84	42	14	700

(续)

c)2 次合并后					
驾驶员年龄	0	1	2	3	合计
30 ~ 65	5635	520	215	130	6500
65 以上	1245	180	60	15	1500
合计	8374	971	447	208	10000

注: * 最不显著检验统计量: 3.86

最不显著检验统计量: 4.99

所有的级别显著地不同

变量的显著性: 0.00001 (Bonferroni 调整)

4.4 CART (分类和回归树)

CART (Classification And Regression Tree, 分类和回归树) 算法由 Breiman 等人[5]于1984年提出。从概念上讲, Breiman 提出的(标称/分类目标变量的)分类树与 ID3 的决策树相同。该算法与 ID3 有三个方面的不同。第一, CART 中用于选择变量的不纯度度量是 Gini 指数 (Gini index)。第二, 如果目标变量是标称的, 并且具有两个以上的类别, 则 CART 可能考虑将目标类别合并成两个超类别。这一过程称为双化 (twoing)。第三, 如果目标变量是连续的 (数值的), 则 CART 算法找出一组基于树的回归方程来预测目标变量。如果目标变量是标称的, 则称该树为分类树 (classification tree), 而对于连续数值目标变量, 则该树称为回归树 (regression tree)。

4.4.1 CART 使用的不纯度度量

有 4 种不同的不纯度度量可用来发现 CART 模型的划分, 这取决于目标变量的类型。对于分类的目标变量, 我们可以选择 GINI、双化或 (对于有序目标) 有序双化。对于连续的目标变量, 我们可以使用最小二乘偏差 (Least-Squared Deviation, LSD) 或最小绝对偏差 (Least Absolute Deviation, LAD)。前三种度量将在以下几节解释。

4.4.2 Gini 指数

Gini 指数是一种不等性度量, 由意大利统计学家 Corrado Gini 提出, 并于 1912 年发表在他的文章“Variabilità e mutabilità”中。它通常用来度量收入不平衡, 但是它可以用来度量任何不均匀分布。Gini 指数是一个 0 ~ 1 之间的数, 其中 0 对应于完全相等 (其中每个人都具有相同的收入), 而 1 对应于完全不相等 (其中一个人具有所有收入, 而其他人收入都为零)。Gini 指数的一个修订形式用于度量节点的不纯度, 并且常常在依赖变量 (目标变量) 是分类变量时使用。它的最小值是 0, 最大值是 $(1 - 1/k)$, 其中 k 是目标变量的类别数。

节点 t 的 Gini 指数 $GINI(t)$ 定义为:

$$GINI(t) = \sum_{j=1}^k p(j/t)p(i/t) \quad (4.3)$$

其中 i 和 j 是目标变量的类别。上式可以写作:

$$GINI(t) = 1 - \sum_j p^2(j/t) \quad (4.4)$$

其中 $p(j/t)$ 表示目标类别 j 在节点 t 中出现的比例。

这样, 当节点中的实例在目标类别之间均匀分布时, Gini 指数取最大值 $1 - 1/k$, 其中 k 是目标变量的类别数。Gini 指数的最小值为 0, 当节点上的所有数据都属于一个目标类别时

取最小值。

节点 t 上的 s 上划分的 Gini 标准定义为:

$$GINI_{split}(s, t) = GINI(t) - p_L \cdot GINI(t_L) - p_R \cdot GINI(t_R) \quad (4.5)$$

其中 p_L 是 t 中送到左边子女节点的实例所占的比例, p_R 是 t 中送到右边子女节点的实例所占的比例。 $s \in S$ 是所有可能的划分集 S 中的一个具体划分。

选择划分 s , 最大化 $GINI_{split}(s, t)$ 的值。由于对于节点 t 上的任意划分 s , $GINI(t)$ 是常量, 我们可以说选择划分 s , 使得量 $Gain(s, t) = p_L \cdot GINI(t_L) + p_R \cdot GINI(t_R)$ 最小。

对于分类变量, 如果类别多于两个, 则考虑将类别合并为两个超类别的所有可能组合, 以求出最佳划分。

4.4.3 使用 Gini 指数——一个例子

让我们用表 4-16 中的气象数据, 使用 Gini 指数构建分类树。变量“天气”和“气温”有三个级别, 因此我们需要考虑级别合并, 得到使用 Gini 指数划分数据的增益。在这个例子中, 我们仅计算三个可能的合并中的一个。读者可以试着计算所有可能的合并, 以确定划分数据的变量。表 4-16 显示了根节点上 Gini 指数的计算结果。计算的细节如下:

节点 0 上的计算:

变量——天气

$$GINI(\text{天气} = \text{晴} + \text{雨}) = 1 - \left[\left(\frac{5}{10} \right)^2 + \left(\frac{5}{10} \right)^2 \right] = \frac{1}{2}$$

$$GINI(\text{天气} = \text{多云}) = 1 - \left[\left(\frac{4}{4} \right)^2 + \left(\frac{0}{4} \right)^2 \right] = 0$$

$$GINI(\text{按天气划分}) = \left[\left(\frac{10}{14} \right) \times \left(\frac{1}{2} \right) + \left(\frac{4}{14} \right) \times 0 \right] = 0.3571$$

变量——气温

$$GINI(\text{气温} = \text{热} + \text{凉爽}) = 1 - \left[\left(\frac{5}{8} \right)^2 + \left(\frac{3}{8} \right)^2 \right] = 0.46875$$

$$GINI(\text{气温} = \text{适中}) = 1 - \left[\left(\frac{4}{6} \right)^2 + \left(\frac{2}{6} \right)^2 \right] = 0.44$$

$$GINI(\text{按气温划分}) = \left[\left(\frac{8}{14} \right) \times 0.46875 + \left(\frac{6}{14} \right) \times 0.44 \right] = 0.456$$

变量——湿度

$$GINI(\text{湿度} = \text{高}) = 1 - \left[\left(\frac{3}{7} \right)^2 + \left(\frac{4}{7} \right)^2 \right] = \frac{24}{49}$$

$$GINI(\text{湿度} = \text{正常}) = 1 - \left[\left(\frac{6}{7} \right)^2 + \left(\frac{1}{7} \right)^2 \right] = \frac{12}{49}$$

$$GINI(\text{按湿度划分}) = \left[\left(\frac{7}{14} \right) \times \left(\frac{24}{49} \right) + \left(\frac{7}{14} \right) \times \left(\frac{12}{49} \right) \right] = 0.37$$

变量——有风

$$GINI(\text{有风} = \text{false}) = 1 - \left[\left(\frac{6}{8} \right)^2 + \left(\frac{2}{8} \right)^2 \right] = 0.375$$

$$GINI(\text{湿度} = \text{true}) = 1 - \left[\left(\frac{3}{6} \right)^2 + \left(\frac{3}{6} \right)^2 \right] = \frac{18}{36} = 0.5$$

$$GINI(\text{按有风划分}) = \left[\left(\frac{8}{14} \right) \times 0.375 + \left(\frac{6}{14} \right) \times 0.5 \right] = 0.43$$

表 4-16 各变量划分根节点上的数据的增益

	天气		气温		湿度		有风	
特征	晴或雨	多云	热或凉爽	温暖	高	正常	FALSE	TRUE
Yes	5	4	5	4	3	6	6	3
No	5	0	3	2	4	1	2	3
Gini(i)	0.5	0	0.47	0.44	0.49	0.25	0.38	0.5
Gain	0.36		0.46		0.37		0.43	

最小增益是按天气划分得到的。因此，在根节点，我们按变量“天气”划分数据。重复这一过程，直到得到纯节点，或满足像叶节点上最小数据量这样的终止条件为止。结果树如图 4-11 所示。

在树中， N 为该节点上的数据点数， V 为将该节点上的数据划分到它的子女节点所基于的变量。

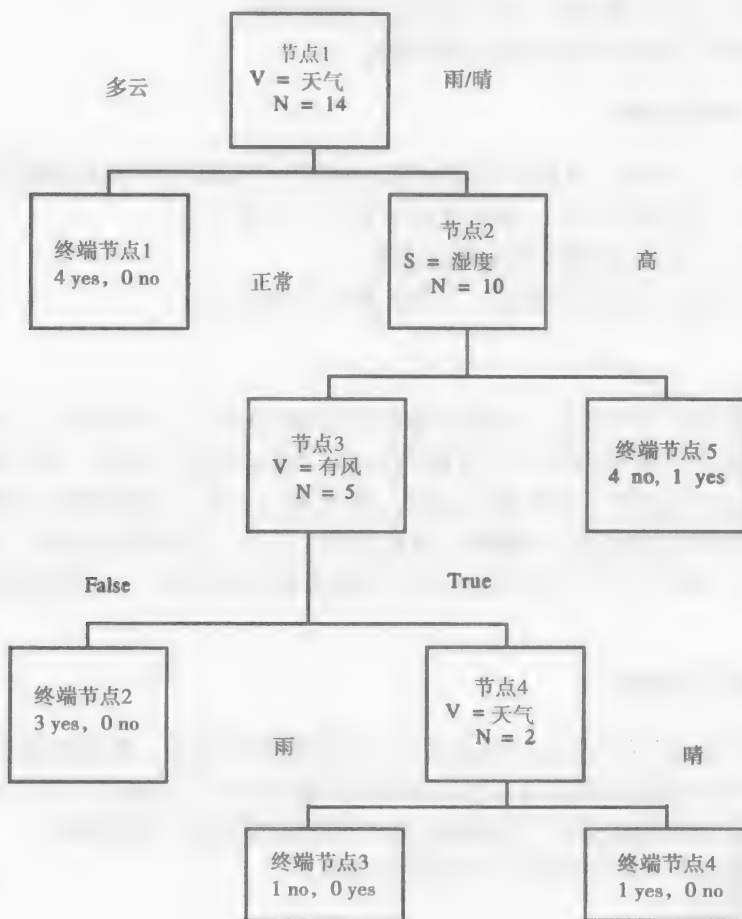


图 4-11 结果树

4.4.4 双化指数

基于双化指数 (twoing index) 将目标类别划分成双超类，然后基于这些双超类找出被预

测变量上的最佳划分。节点 t 上的 s 上划分的双化标准定义为:

$$GINI_{\text{Twoing}}(s, t) = \frac{P_L \cdot P_R}{4} \left[\sum_j \left| p\left(\frac{j}{t_L}\right) - p\left(\frac{j}{t_R}\right) \right| \right]^2 \quad (4.6)$$

其中 t_L 和 t_R 是被划分 s 创建的节点。选取最大化该标准的划分作为划分 s 。这里, 下标 j 遍取被预测变量的类别(基于变量的类别来划分数数据)。在分类目标变量情况下, 可以用目标类别的各种组合创建超类 C_1 和 C_2 。

4.4.5 有序双化

有序双化指数是双化指数的修改, 用于有序目标变量。二者的区别在于, 使用有序双化标准, 只能合并相邻的类别以形成超类。例如, 考虑一个诸如账户状态这样的目标变量, 类别 1 = 当前, 类别 2 = 30 天过期, 类别 3 = 60 天过期, 类别 4 = 90 天以后过期。双化标准可能在某种情况下将类别 1 和 4 合并在一起, 形成超类。然而, 如果认为这些类别是有序的, 则我们不希望合并类别 1 和 4 (外部还包括介入类别), 因为它们不是相邻的。有序双化指数考虑这种序, 并且不合并像类别 1 和 4 这样的不相邻类别。

最后, 我们在下面概述 CART 的分类树算法。

4.4.6 CART 分析的步骤

1) 从根节点 $t=1$ 开始, 从所有可能候选 s 的集合中搜索使不纯度降低最大的划分 s^* 。然后, 使用划分 s^* 将节点 1 ($t=1$) 划分成两个节点 $t=2$ 和 $t=3$ 。

2) 在 $t=2$ 和 $t=3$ 上分别重复划分搜索过程。

继续树生长过程, 直到至少满足一个停止树生长规则为止。

4.5 回归树

多元回归分析是一个老工具。按照 Draper 和 Smith (1980) [5] 的说法, 它可以追溯到英国人类学家和气象学家弗朗西斯·高尔顿 (Francis Galton) 爵士 (1822—1911) 的工作。回归分析可以松散地定义为研究一个依赖 (或响应) 变量和一组独立 (或预测子) 变量之间关系的方法的应用。这种研究通常基于对象集上测量的样本。这一过程与决策树一节介绍的监督学习框架非常吻合。在 4.5.1 节, 我们将介绍一种简单回归任务和一个由回归树学习方法得到的可能的模型。

4.5.1 回归树的一个例子

Harrison 和 Rubinfeld (1978) [6] 描述了一个有趣的回归应用。在该应用中, 他们试图使用其他变量预测波士顿地区的房价。他们的目标是检查空气污染浓度 (NOX) 是否对房价有影响。他们收集的数据包括 506 个观测值, 每个观测值都用如下变量描述:

- MV (目标变量): 房价中值 (以千美元为单位)。
- CRIM: 犯罪率。
- ZN: 特殊用途土地所占百分比。
- INDUS: 非零售业百分比。
- CHAS: 如果在 Charles 河为 1, 否则为 0。
- NOX: 氮氧化物浓度 (ppm)。

- RM: 平均房间数。
- AGE: 1940 年前建造的房屋所占百分比。
- DIS: 到就业中心的加权距离。
- RAD: 到干线公路的可达性。
- TAX: 税率。
- P/T: 小学生/教师比。
- B: 黑人所占百分比。
- LSTAT: 低地位人口所占百分比。

根据这 506 个案例, 我们可以得到一个回归模型, 从而捕捉房价对其他变量的依赖性。这个模型不仅可以作为未来案例的预测工具, 而且也将充实我们关于变量之间关系的知识。如果决定使用回归树学习方法, 则我们可以得到图 4-12 给出的基于树的模型。

回归树的图形表示具有两类节点。圆形节点表示输入变量上的测试, 而方框是树叶, 表示目标变量的预测。使用该模型预测可以通过如下方法得到: 按照输入变量上测试的输出, 让检验案例沿着正确的分支在树上“下落”。案例到达的树叶给出对应的预测。

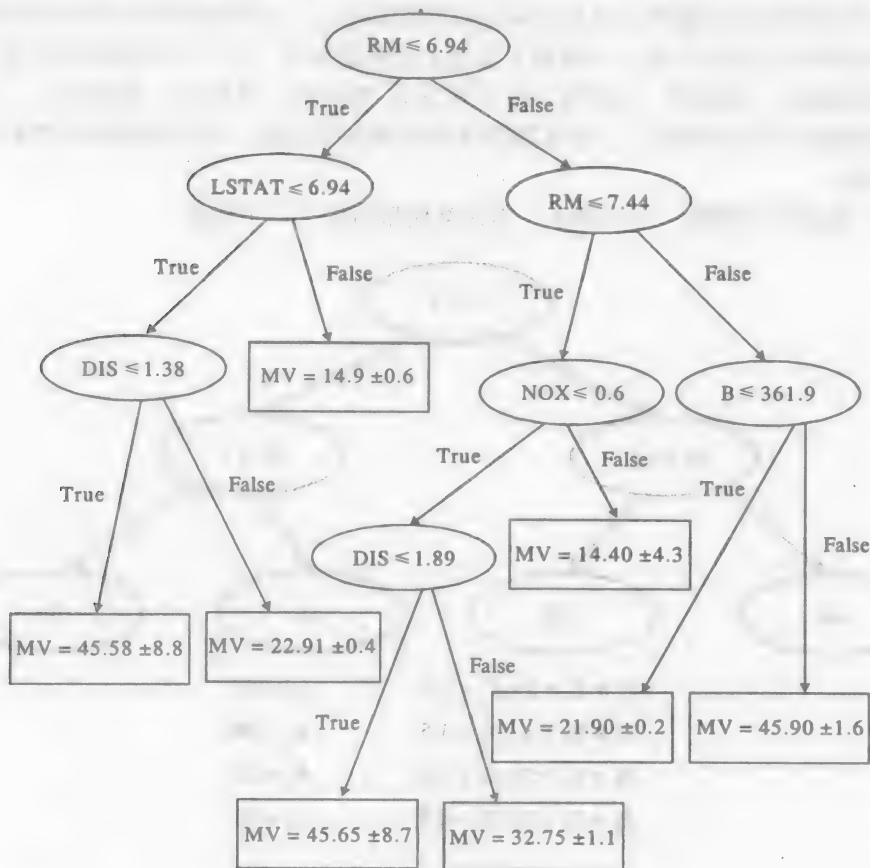


图 4-12 波士顿住宅数据的回归树

4.5.2 基于树的回归

基于树的回归模型的研究可追溯到 Morgan 和 Sonquist[7] (1963) 和他们的 AID (Automat-

ic Interaction Detection, 自动交互检测) 程序。然而, 这方面研究的主要引文一直是 Breiman 和他的同事的关于分类与回归树的原创性著作 (1984)。这些作者给出了基于树模型的分类与回归的全面介绍。在处理具有大量变量和案例的领域时, 基于树的回归模型以其简洁性和有效性而闻名。回归树使用快速的分治贪心算法来构建, 该算法递归地将给定的训练数据划分成较小的子集。这种算法的使用是这些方法有效的原因, 然而, 由于基于小案例样本估计的不可靠性, 它也可能导致树的较低层上的决策质量较低。处理这些问题的方法几乎与初始树的生长一样重要。回归树可以看作如下形式的一类加法模型 [8, 9, 10, 11] (Hastie 和 Tibshirani, 1990):

$$m(x) = \sum_{i=1}^I k_i \times I(x \in D_i) \quad (4.7)$$

其中 k_i 是常量; $I(\cdot)$ 是一个指示函数, 如果参数为真返回 1, 否则返回 0; D_i 是训练数据 D 的不相交划分, 使得 $\bigcup_{i=1}^I D_i = D$ 并且 $\bigcap_{i=1}^I D_i = \phi$ 。

这类模型有时称作分段常量回归模型 (piecewise constant regression model), 因为它们将预测子空间 \mathcal{X} 划分成区域集合, 并在每个区域拟合一个常量值。基于树的回归模型的一个重要特点是它们用树的形式提供了这些区域的前置逻辑表示。由树根到树叶的每条路径对应一个区域。树的每个内部节点是一个预测子变量上的测试逻辑。在二元树的特殊情况下, 测试有两个可能的输出: 真或假。这意味与每个划分 D_i 相关联, 我们有一条路径 P_i , 它由预测变量上的逻辑测试的合取组成。当我们想更好地理解回归面时, 回归函数的这种符号表示是一个重要问题。

图 4-13 通过回归树的一个小例子, 为这类模型给出了一个说明。

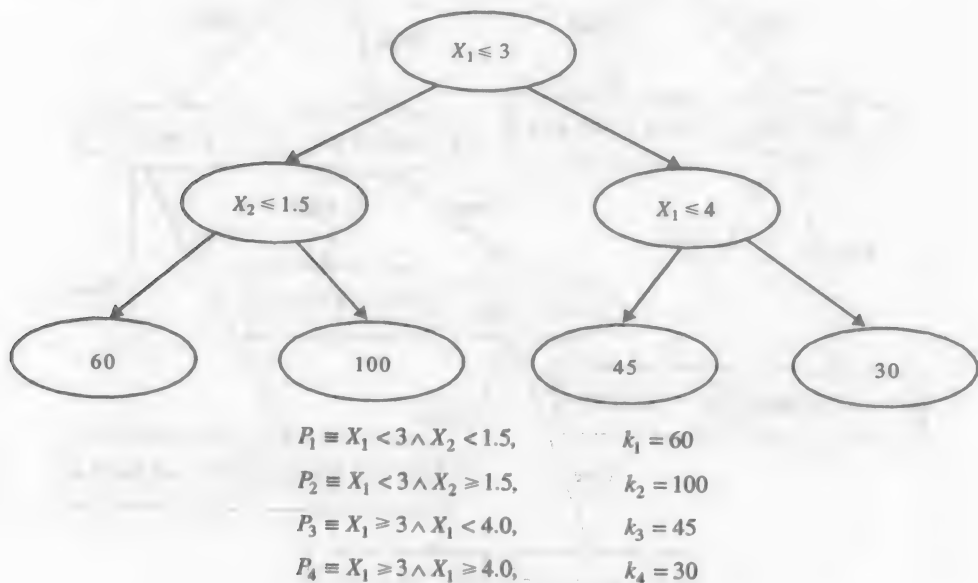


图 4-13 回归树

由于从根节点到树叶有 4 条不同路径, 因此这棵树将输入空间划分成 4 个不同的区域。如前所示, 每条路径上测试的合取可以看作这些区域的一个逻辑描述。该树粗略地对应于图 4-14 所示的回归面。(假定只有两个预测变量 X_1 和 X_2 。)

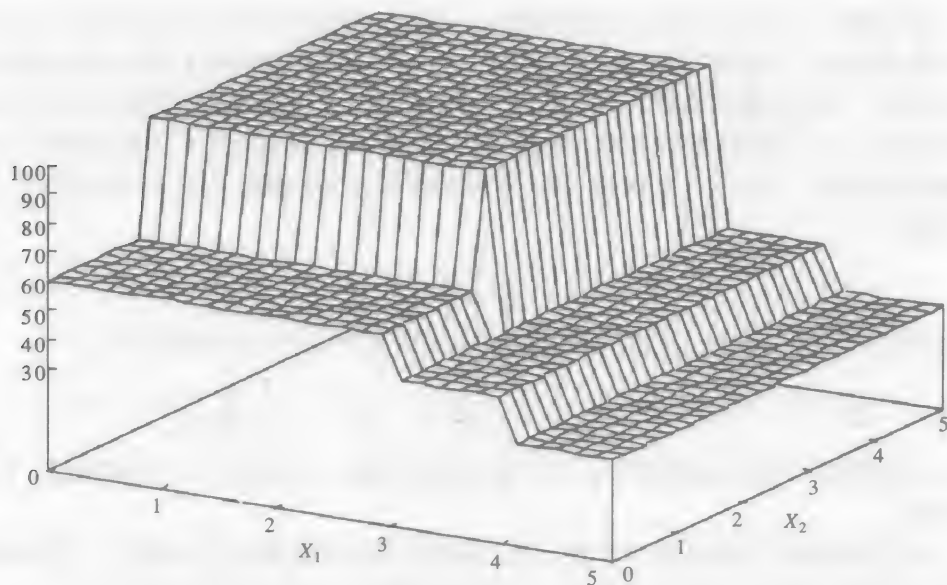


图 4-14 回归面

使用式(4.7)更简洁地表示, 我们得到:

$$m(x) = 60 \times I(X_1 < 3 \cap X_2 < 1.5) + 100 \times I(X_1 < 3 \cap X_2 \geq 1.5) + 45 \times I(X_1 \geq 3 \cap X_2 < 4) + 30 \times I(X_1 \geq 3 \cap X_2 \geq 4)$$

回归树使用递归划分(RP)算法构建。该算法通过递归地将训练样本划分成较小的子集来构建树。我们将粗略介绍该算法。RP算法接受 n 个数据点的集合作为输入, 并且当特定终止条件不满足时, 它就产生一个测试节点 t , 其分支通过对输入数据点的两个子集使用相同的算法得到。这两个子集分别由节点 t 上的逻辑上满足划分测试 s^* 的案例和其余案例组成。在每个节点, 根据某种局部标准, 选择最好的划分测试。这意味着是一种贪心的爬山算法。

4.5.3 最小二乘方回归树

大部分常用的基于未知回归面样本的构造回归模型的方法都试图得到最小化最小平方误差的模型参数:

$$\frac{1}{n} \sum_{i=1}^n (y_i - r(\beta, x_i))^2 \quad (4.8)$$

其中 n 是样本大小, $\langle x_i, y_i \rangle$ 是一个数据点, 而 $r(\beta, x_i)$ 是回归模型 $r(\beta, x)$ 对案例 $\langle x_i, y_i \rangle$ 的预测。

如果叶值(节点值)取常数, 则应当赋予使用最小平方误差标准得到的回归树的树叶的常量是每个树叶 l 中案例的目标值的平均值。

$$k_l = \frac{1}{n_l} \sum_{i \in D_l} y_i \quad (4.9)$$

其中 n_l 是包含在树叶 l 中的案例的集合 D_l 的基数(即 $n_l = \#D_l$)。

有些系统, 如 RETIS(Karalic, 1992)和 M5(Quinlan, 1992), 在树叶中使用其他非常量模型。它们使用线性多项式, 而不是平均值。

关于划分规则,我们只介绍二元树的情况。这些树的每个内部节点具有两个后代节点。这些内部节点根据一个输入变量上的测试结果,将训练实例划分成两个子集。满足测试的案例沿着左分支,而其他案例沿着右分支。划分测试选择的目标是改进结果树的拟合误差。从根节点到节点 t 的一条路径对应于输入案例的一个划分 D_t 。假定用式4.9得到常数,由最小平方误差标准得到,我们定义节点 t 的拟合误差为该节点中实例的 Y 值与节点常量 k_t 的平方差的平均值。

$$Err(t) = \frac{1}{n_t} \sum_{D_t} (y_i - k_t)^2 \quad (4.10)$$

其中 k_t 由式4.9定义。此外,我们定义树 T 的误差为其叶节点误差的加权平均:

$$Err(T) = \sum_{l \in \tilde{T}} P(l) \times Err(l) = \sum_{l \in \tilde{T}} \frac{n_l}{n} \times \frac{1}{n_l} \sum_{D_l} (y_i - k_l)^2 = \frac{1}{n} \sum_{l \in \tilde{T}} \sum_{D_l} (y_i - k_l)^2 \quad (4.11)$$

其中 $p(l)$ 是案例落入树叶 l 的概率, n 是训练案例的总数, n_l 是树叶 l 中的案例数, \tilde{T} 是树 T 的树叶集合。

二元划分将案例集划分成两个子集。划分规则的目标是选择这样的划分:它使该划分导致的树的误差减到最低。我们定义划分 s 的误差为结果子节点误差的加权平均:

$$Err(s, t) = \frac{n_{t_L}}{n_t} \times Err(t_L) + \frac{n_{t_R}}{n_t} \times Err(t_R) \quad (4.12)$$

现在,我们已经准备就绪,可以定义给定候选划分集 S 时,节点 t 的最佳划分。

定义4.1 最佳划分 s^* 是属于 S 的划分,它使下式最大化:

$$\Delta Err(s, t) = Err(t) - Err(s, t)$$

这个贪心标准指导 LS 回归树所有内部节点的划分选择。在 RP 算法的每次迭代中,评估每个预测子变量的所有可能划分,并选取具有最好 $\Delta Err(s, t)$ 的划分。

关于树生长方法的最后一个问题(即停止规则),关键问题是用于选择划分的误差评估的可靠性。前面介绍的所有误差度量都是统计意义下的估计,因为它们都是训练样本的函数(通常称作再代入估计)。这些估计的准确率(accuracy)在很大程度上依赖于样本的质量(quality)。随着算法递归地划分原训练集,划分使用越来越小的样本评估。这意味随着树的生长,估计变得越来越不可靠。容易证明,在树生长期间, ΔErr (见定义4.1)总是大于或等于0。显然,我们总是得到越来越精确的回归树模型。考虑一个极端情况:一棵过分大的树,每个树叶只有一个训练案例,它将具有零误差。

这种推理的问题恰恰是由于得到估计的训练案例量的不足而导致的估计的可靠性问题。基于小样本的估计很难泛化到未知案例,因此导致预测准确率很差的模型。这通常称为过分拟合训练数据(overfitting the training data)。

有两个替代过程可将该问题最小化。第一个是制定一个可靠性标准,确定何时应当停止树的生长。在基于树的模型中,这通常称为先剪枝(pre-prune)。第二个,也是最常使用的过程是产生一棵非常大的(不可靠的)树,然后对它进行后剪枝(post-prune)。回归树的剪枝是得到准确的树的基本步骤,这将在4.7节中介绍。使用后剪枝方法,停止条件通常是非常“宽松的”,因为还有一个后剪枝阶段。其基本思想是不能因为过早地停止初始生长阶段而“丢失”任何可能的、好的后剪枝树。一种经常使用的标准是对最小案例数加以限制,一旦达到就强制终止 RP 算法。停止标准的另一个例子是,如果当前节点的误差率低于根节点的误差率的某一比例就创建一个树叶。

4.5.4 LS 回归树的有效生长

用于生长回归树的递归划分(RP)算法的计算复杂性在很大程度上依赖于给定节点的最佳划分的选择。这一任务重新尝试每个输入变量的所有可能划分。一个变量的可能划分数依赖于它的类型。下面将给出用于生长LS回归树的RP算法的更详细版本:

算法 4.5.1 生长LS回归树。

输入: n 个数据点的集合 $\{ \langle x_i, y_i \rangle \}, i=1, \dots, n$

输出: 一棵回归树。

方法:

```

IF 满足终止标准 THEN
    创建一个树叶节点, 并将  $n$  个数据点的平均  $y$  值赋予它
    Return 树叶节点
ELSE
     $s^* = \langle \text{任意划分} \rangle$ 
    FOR 所有变量  $X_V$  DO
        IF  $X_V$  是标称变量 THEN
            最佳划分  $X_V = \text{TryAllNominalSplits}(\{ \langle x_i, y_i \rangle \}, X_V)$ 
        ELSE IF  $X_V$  是数值变量 THEN
            最佳划分  $X_V = \text{TryAllNumericSplits}(\{ \langle x_i, y_i \rangle \}, X_V)$ 
        ENDIF
        IF 最佳划分  $X_V$  比  $s^*$  好 THEN
             $s^* = \text{最佳划分 } X_V$ 
        ENDIF
    ENDFOR
    用  $s^*$  创建节点  $t$ 
    Left_branch( $t$ ) = GrowLStree( $\{ \langle x_i, y_i \rangle : x_i \rightarrow s^* \}$ )
    Right_branch( $t$ ) = GrowLStree( $\{ \langle x_i, y_i \rangle : x_i \rightarrow s^* \}$ )
ENDIF

```

该算法的主要计算量是尝试变量的所有可能划分。评估每个试验划分, 这意味需要得到结果子节点模型, 以便计算它们的误差(比较式(4.10)和式(4.12))。假定常量模型由式(4.9)定义, 我们需要(对划分的每个分支)计算两个平均值, 以评估一个划分(定义4.1)。式(4.10)实际上类似于计算变量方差的公式。这一计算涉及扫描数据两次, 一次得到平均值, 而第二次计算平方差。这一开销可以通过使用如下等价公式降低

$$Err(t) = \frac{\sum_{D_t} y_i^2}{n_t} - \left(\frac{\sum_{D_t} y_i}{n_t} \right)^2 \quad (4.13)$$

该计算可以通过单遍扫描数据完成。即便使用这个公式, 评估每个试验划分的开销仍然为 $O(n_t^2)$ 。使用允许增量计算一个变量的所有划分的化简, 可以降低这一开销。根据定义4.1给出的公式, 最佳划分 s^* 是最小化式(4.12)给定的值的划分。使用式(4.13)中的公式, 我们得到

$$Err(s, t) = \frac{n_{t_l}}{n_t} \times \left(\frac{\sum_{D_{t_l}} y_i^2}{n_{t_l}} - \left(\frac{\sum_{D_{t_l}} y_i}{n_{t_l}} \right)^2 \right) + \frac{n_{t_r}}{n_t} \times \left(\frac{\sum_{D_{t_r}} y_i^2}{n_{t_r}} - \left(\frac{\sum_{D_{t_r}} y_i}{n_{t_r}} \right)^2 \right) \quad (4.14)$$

为了简化记号, 令 SS_L 和 SS_R 分别等于 $\sum_{D_{t_l}} y_i^2$ 和 $\sum_{D_{t_r}} y_i^2$, 并且 S_L 和 S_R 分别等于 $\sum_{D_{t_l}} y_i$ 和 $\sum_{D_{t_r}} y_i$, 得到

$$Err(s, t) = \frac{SSL}{n_t} - \frac{S_L^2}{n_t n_t} + \frac{SS_R}{n_t} - \frac{S_R^2}{n_t n_t} \quad (4.15)$$

容易看出, 无论被评估的划分是什么, 该公式的第一项是常量。这是因为

$$D_t = D_{t_L} \cup D_{t_R}$$

因此

$$\sum_{D_{t_L}} y_i^2 + \sum_{D_{t_R}} y_i^2 = \sum_{D_t} y_i^2 \quad (4.16)$$

这意味 $SSL + SSR$ 总是常量, 并且等于 $\sum_{D_t} y_i^2$ 。

这意味不同候选划分之间的唯一差别在于最后一项。这种化简对于评估和选择节点上的最佳划分的方法具有重要意义。使用这些结果, 我们可以得到变量的最佳划分 s^* 的新定义, 与前面的定义(定义 4.1)相比, 它在计算效率方面具有明显优势。注意, 仅当定义式(4.9)给定的常量模型(即假定最小平方误差)时这种化简才是有效的。由于我们的目标是最小化先前导出的表达式, 因此得到节点最佳划分的新定义。

定义 4.2 最佳划分 s^* 是属于 S 的划分, 它使下式最大化

$$\frac{S_L^2}{n_{t_L}} + \frac{S_R^2}{n_{t_R}} \quad (4.17)$$

其中:

$$S_L = \sum_{D_{t_L}} y_i \text{ and } S_R = \sum_{D_{t_R}} y_i$$

正如我们在下面几节将看到的, 这个定义可以对任意预测变量的所有候选划分 S 快速、增量地求值。

4.5.5 连续变量上的划分

现在, 我们给出一个算法, 它使用定义 4.2 的结果找出连续变量的最佳划分。假定我们有一个具有 n_t 个案例的集合, 其 y 值和为 S_t 。算法 4.5.2 得到连续预测变量 X_v 上的最佳划分。

算法 4.5.2 找出连续变量的最佳划分。

输入: n_t 个案例, 它们的 Y 值和(S_t), 变量 X_v

输出: X_v 上最佳割点划分

方法: 按 X_v 上的值对案例排序

$S_R = S_t; S_L = 0;$

$n_R = n_t; n_L = 0;$

BestTillNow = 0

FOR 所有的实例 i DO

$S_L = S_L + Y_i; S_R = S_R - Y_i$

$n_L = n_L + 1; n_R = n_R - 1$

IF $(X_{i+1,v} > X_{i,v})$ THEN % 如果值相等, 则无试验

NewSplitValue = $(S_L^2/n_L) + (S_R^2/n_R)$

IF (NewSplitValue > BestTillNow) THEN

BestTillNow = NewSplitValue

BestCutPoint = $(X_{i+1,v} + X_{i,v}) / 2$

ENDIF

ENDIF

ENDFOR

(这里, $X_{i,v}$ 表示变量 X_v 排序后的第 i 个值)

这个算法主要有两个部分。第一部分按所考虑的变量的值对实例进行排序。使用快速排序, 其平均复杂度为 $O(n_i \log n_i)$ 。对于以有效的方式快速试验所有的割点值, 排序操作是必要的。我们只需要尝试这些割点值, 因为只有它们才可能改变定义 4.2 中的分值, 因为它们改变了分到左和右分支的案例集。算法的第二个相关部分是所有候选划分的评估。试验划分分数最多为 $n_i - 1$ (如果变量 X_v 的所有变量的值都不相同)。如果没有定义 4.2 给定的等式, 我们必须计算候选划分的每个划分的“方差”。

4.5.6 离散变量上的划分

标称(或离散)变量上的划分通常涉及尝试所有可能的形如 $X_v = x_v$ 的检验, 其中 x_v 是变量 X_v 的一个可能的值。如果有许多可能的值, 则通常生成较大的树。另一种方法是不用二叉树, 而是对变量的每个可能值产生一个分支。这会增加训练样本划分, 这导致不可靠估计的速度比二元划分更快。另一种选择是考虑形如 $X_v \in \{x_v \dots\}$ 的测试。这种方法需要附加的计算开销, 尽管它可以改善结果树的可理解性, 并且不会对训练案例进行过度划分。Breiman 等(1984)证明了一个有趣结果。

Breiman 等(1984)建议的方法涉及一个初始阶段, 其中将节点的实例排序如下:

假定 B 是 X_v 的值的集合, 它出现在当前节点 t 中(即 $B = \{b: x_i \in t \wedge x_{i,v} = b\}$), 并定义 $\bar{y}(b_i)$ 为变量 X_v 上的值等于 b_i 的实例的 Y 的平均值, 我们对这些值排序, 使得

$$\bar{y}(b_1) \leq \bar{y}(b_2) \leq \dots \leq \bar{y}(b_{\#B})$$

$$X_v \in \{b_1, b_2, \dots, b_{\#B}\}, h = 1, 2, \dots, \#B - 1$$

变量的值按这种方法排序后, Breiman 和他的同事证明了节点 t 中离散变量 X_v 上的最佳划分是这 $\#B - 1$ 个划分中的一个。

这个定义源于 Fisher(1958)对回归的最小二乘方误差标准证明的一个定理, 并被 Breiman 和他的同事(1984, 9.4 节)推广到较大的凹的不纯(误差)函数类。Chou(1991)进一步将这些结果推广到任意个箱(即不限于二元划分)和其他误差函数。使用这种方法, 我们只需要考察 $\#B - 1$ ($\#$ 表示集合中元素的个数)个而不是 $2\#B - 1$ 子集。注意, 我们仍需要“扫描”所有数据, 得到值 $y_i b_i$, 加上 $\#B$ 个元素排序操作。在给出离散划分算法之前, 我们用一例子解释这种方法。

例 4.3 假设我们在节点 t 有如表 4-17 所示的实例, 得到平均值

$$\bar{y}(\text{绿}) = (24 + 29 + 13)/3 = 22$$

$$\bar{y}(\text{红}) = (56 + 45)/2 = 50.5$$

$$\bar{y}(\text{蓝}) = (120 + 100)/2 = 110$$

如果我们按照对应的 Y 值对这些值排序, 我们得到 {绿, 红, 蓝}。根据 Breiman 定理, 最佳划分将是 $\#B - 1$ (这里为 $3 - 1 = 2$) 个划分中的一个, 即 $X_v \in \{\text{绿}\}$ 和 $X_v \in \{\text{绿, 红}\}$ 之一。

表 4-17 例子 4-3 的实例

颜色	Y
绿	24
红	56
绿	29
绿	13
蓝	120
红	42
蓝	100

按前面介绍的方法对实例排序后, 我们使用如下增量算法, 它类似于连续变量的算法。

算法 4.5.3 找出离散变量的最佳划分。

输入: n_i 个案例, 它们的 y 值和 (S_i) , 变量 X_v 。

输出: X_v 值的有序集合和该集合的一个划分。

方法:

得到与 X_v 的每个值相关联的 Y 平均值
按照与 X_v 的每个值相关联的 Y 平均值对 X_v 值排序

$S_R = S_t; S_L = 0;$

$n_R = n_t; n_L = 0;$

BestTillNow = 0

FOR 得到的有序值集的每个值 b DO

$YB = X_v$ 等于 b 的案例的 Y 值和

$NB = X_v$ 等于 b 的案例数

$S_L = S_L + YB; S_R = S_R - YB$

$n_L = n_L + NB; n_R = n_R - NB$

$\text{NewSplitValue} = S_L = (S_L^2/n_L) + (S_R^2/n_R)$

IF ($\text{NewSplitValue} > \text{BestTillNow}$) THEN

BestTillNow = NewSplitValue

BestPosition = 有序值集中 b 的位置

ENDIF

ENDFOR

4.5.7 模型树

模型树推广了回归树的概念。回归树在其每个级别具有常量值 (Written 和 Frank, 2000) [12]。因此它们类似于分段线性函数 (因此是非线性的)。回归树的计算量随着维度的增加而迅速增加。模型树能够有效地学习, 并且可以处理多达数百属性的很高的维度 (见图 4-15)。与回归树相比, 模型树的主要优点是比回归树小得多, 决策能力是明显的, 并且回归函数一般并不涉及许多变量。

一个称作 M5 算法 (M5 algorithm) 的算法用于归纳模型树 (Quinlan, 1992), 其过程如下:

假设可用的训练实例的集合为 T 。每个实例用一组固定 (输入) 属性的值刻画, 并有一个相关联的目标 (输出) 值。目标是构造一个模型, 将训练案例的目标值与输入属性值联系起来。模型的质量通常用它们预测未知案例目标值的准确率来度量。特征空间的回归树如图 4-16 所示。

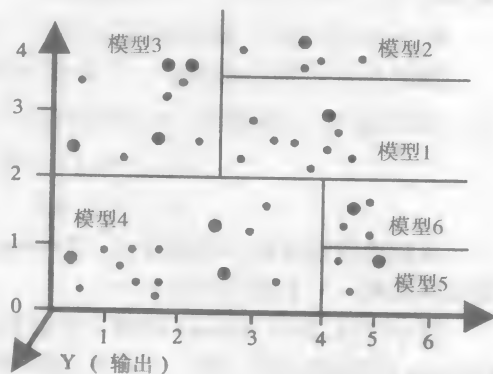


图 4-15 回归模型

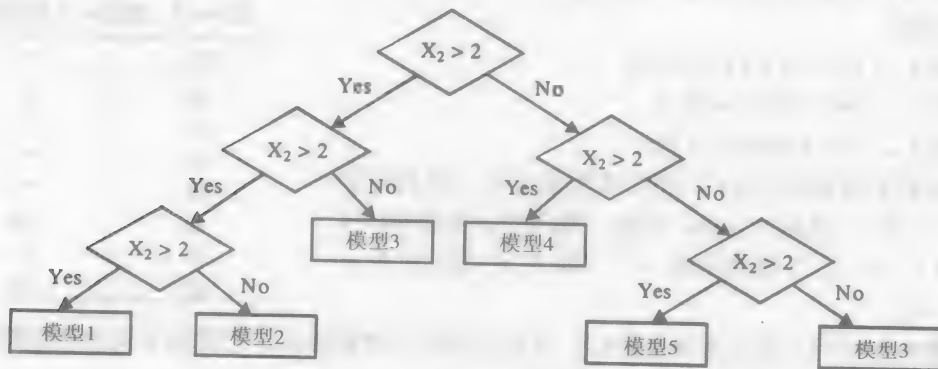


图 4-16 模型回归树

基于树的模型采用分治方法构造。集合 T 或者与一个树叶相关联, 或者选择某个测试, 根据测试的输出将 T 划分成子集, 并且对这些子集递归地应用相同的过程。M5 模型树算法的划分标准是, 将到达一个节点的类值的标准差看作该节点误差的度量, 并计算该误差的期望减少作为该节点上每个属性的检测结果。计算标准差减少 (SDR) 的公式是:

$$SDR = sd(T) - \sum_i \frac{|T_i|}{|T|} sd(T_i) \quad (4.18)$$

其中, T 表示到达该节点的实例的集合, T_i 表示具有第 i 个可能输出的实例子集, sd 表示标准差。

考察所有可能的划分之后, M5 选择期望误差减少最大的划分。当到达一个节点的实例只有少许不同或者只剩下少量实例时, M5 的划分停止。不断地划分通常产生过分复杂的结构, 因此必须剪枝。例如, 用一个树叶替换一棵子树。在最后阶段, 使用光滑过程, 抵消剪枝后的树叶上邻近线性模型之间不可避免地出现的尖峰不连续性。对于用较少训练实例构造的某些模型, 这一步尤其重要。在光滑时, 用这样的方式修改相邻的线性方程, 使得对应于不同方程的相邻的输入向量的预测输出值变得接近。这一过程的细节可以在 Quinlan (1992) 中找到, 并且由 Witten 和 Frank (2000) 给出。

M5 模型树的水文学应用的一个例子如下:

```
If Qt ≤ 51.2:
    and if Qt ≤ 28.7: LM1 (903/5.66%)
        Else if Qt > 28.7: LM2 (379/13.1%)
            Else Qt > 51.2: LM3 (572/66.7%)
```

下面是产生的线性模型:

$$LM1: Q_t + 3 = -0.0118 + 0.317RE_t + 0.124RE_t - 1 + 0.0844RE_t - 2 - 0.109RE_t - 3 \\ + 1.09Q_t - 0.0826Q_t - 1$$

$$LM2: Q_t + 3 = -0.262 + 11.9RE_t + 0.182RE_t - 1 + 8.9RE_t - 2 - 0.198RE_t - 3 + 3.66Q_t \\ - 0.0826Q_t - 267Q_t - 1$$

$$LM3: Q_t + 3 = 15.5 + 25.7RE_t + 7.59RE_t - 1 - 0.0923RE_t - 3 + 1.44Q_t - 0.732Q_t - 1$$

4.6 具有未知类值数据的类预测的一般问题

决策树的一个作用是预测类未知的数据集的类值。当我们使用它做重要决策时, 应当知道可以在多大程度上依赖决策树预测的类值。或者说, 我们的决策树对未来数据集的预测效果如何。为了理解依赖我们构造的树进行决策所涉及的风险, 我们将从不同的角度考察学习问题。

假定给学习算法 (决策树算法) 提供一系列形如 $[X_1, f(X_1)]$, $[X_2, f(X_2)]$, \dots 的训练实例。学习算法返回一个逼近目标函数 f 的假设 h 。为了进一步简化, 假定我们有如图 4-17 所示的训练实例, 假定函数 $f(x)$ 是连续变量。

假设我们 (使用不同的学习算法) 得到图 4-18 所示的假设。

我们做出如下推断:

假设 G 太简单——只有两个点接近该直线。

假设 R 和 B 逼近训练案例一样好, 但在如何对未知输入赋值方面有差别。由于 f 是未知的, 因此没有理由认为 B 比 R 好。

一般来说, 如果两个假设逼近 f 的效果一样好, 则我们没有先验 (piori) 理由认为一个比另一个好。



图 4-17 待学习的数据集

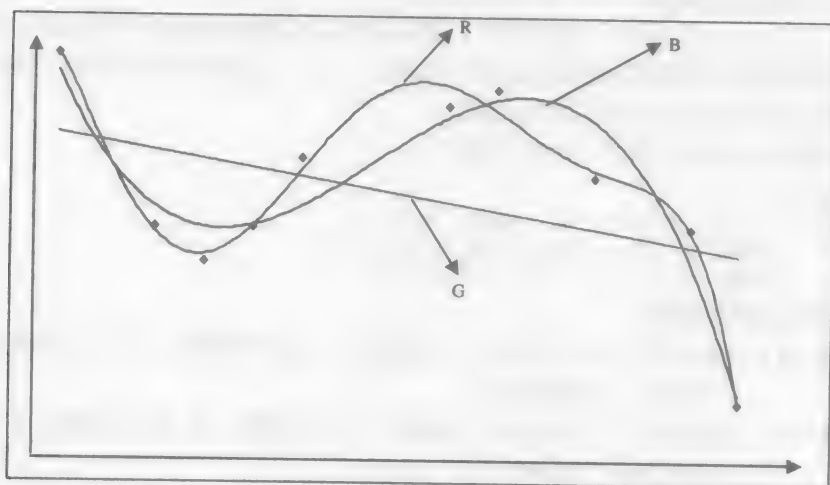


图 4-18 不同算法得到的假设

科学界有一个由来已久的传统：在其他条件相同的情况下，简单的理论比复杂的理论更可取。这被称作 Occam 剃刀 (Occam's Razor)，以中世纪哲学家 William Occam 的名字命名。Occam 剃刀剃掉理论的哲学毛发。其基本思想是：最好的科学理论是揭示所有事实的最简单理论。正如爱因斯坦的名言所说“应该是每件事都尽可能简单，但不是更简单。”当然，许多事情都隐藏在“其他条件相同”中，并且很难客观地评估某个理论是否确实“解释”了它所基于的所有事实——这正是科学界在争论的一个问题。

所有表现出偏爱的学习算法都呈现归纳偏倚 (inductive bias)。Occam 剃刀偏爱拟合数据的最简单假设。这种偏倚将更喜欢假设 B。如果没有某种类型的归纳偏倚，泛化将是不可能进行的。没有泛化，我们就不可能决定如何对以前从未遇到的情况做出反应。

为了进一步理解泛化概念，我们考虑另一个简单例子。考虑图 4-19 所示的数据。假设 A 通过所有的点。假设 B 并未通过所有的点，但是它确实抓住了数据的一般趋势。在这里，我们宁愿用假设 B 来预测未知数据 (其 $f(x)$ 值未知的 x) 的函数值。假设 A 过分拟合数据。如何避免过分拟合？在决策树的情况下，我们能够总是划分节点，直到得到纯节点。如果我们这样做，树就变得非常大，并且过分拟合数据。这时，问题是何时停止分裂节点？这个问题导致了剪枝概念的产生。

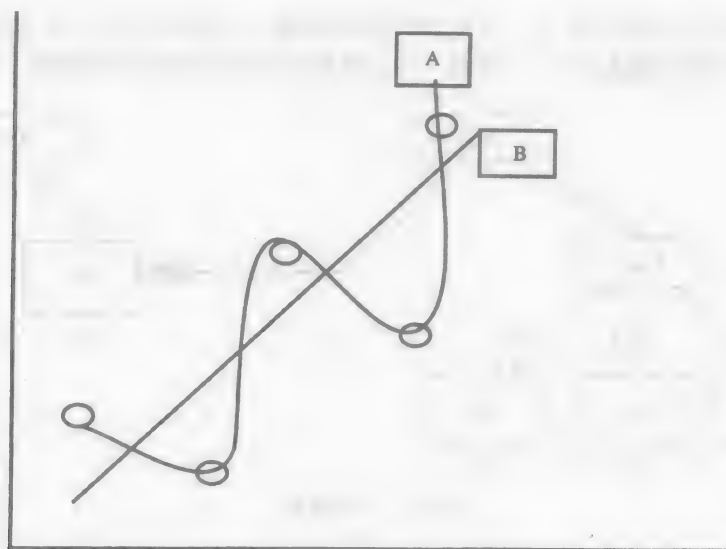


图 4-19 理解泛化和拟合数据误差

4.7 剪枝导论

4.1 节到 4.6 节介绍的方法使用一种递归划分训练集的算法得到一棵树。这样做的结果是，随着树的生长，最佳划分的选择基于越来越小的样本来进行。树的较低层上划分选择通常会变得统计上不可靠，尽管基于训练数据的误差估计（所有节点中误分类的数据总数在数据点总数中所占的比例）持续降低。通常不太可能认为这种误差估计可以泛化到未见过的案例上，并且称树过分拟合训练数据。这意味树捕获了训练样本的规律，而不是得到样本的领域（总体）的规律。这就是修剪树模型的动机。然而，正如 Schaffer[13]所指出的，剪枝不可能视为改善树预测误差的统计手段。事实上，很容易找到一个现实世界领域，剪枝对于独立的、大量检验样本而言，会降低预测准确率。相反，正如 Schaffer[13]所建议的，剪枝应当被视为优先选择较简单的模型。理解不同剪枝方法的偏倚将对选择最适合用户偏爱的策略提供有用的提示。

后剪枝是一个过程，通过该过程产生一棵大树，然后使用可靠的评估方法选择对初始模型而言“尺寸合适的”剪枝后的树。后剪枝方法是计算低效的，即通常可以找到一个领域，其中具有数千个节点的大树经过后剪枝得到具有数百个节点的树。显然这会造成计算上的浪费。一种替代的方法是，一旦进一步划分被认为是不可靠的，就尽快停止树的生长。这就是所谓的树的先剪枝。与后剪枝相比，先剪枝具有明显的计算优势。事实上，我们可以较早地停止树的生长，并且还可以避免后剪枝。然而，过早地停止树的生成会使这种方法面临选择次最优树的危险（Breiman 等，1984）[1]。正因为如此，通常避免过分拟合的方法是后剪枝。

对于后剪枝，已经考虑了两种大不相同的操作：子树置换（subtree replacement）和子树提升（subtree raising）。在每个节点，学习方案可以决定是应该进行子树置换、子树提升，还是保留子树不剪枝。

子树置换：子树置换是主要的剪枝操作，我们首先来介绍它。其基本思想是选择某些子树，并用单个树叶置换它们。例如，图 4-20 中的整个子树涉及一个内部节点和 3 个叶节点，它被一个类值为“*Yes*”的树叶所置换。这可能导致训练集的准确率下降，因为如果原来的树

由4.1节~4.4节介绍的算法产生,算法将继续构造树,直到所有的叶节点是纯的为止(或者直到所有的属性都被测试为止)。然而,它可能提高在独立选取的检验集上的准确率。

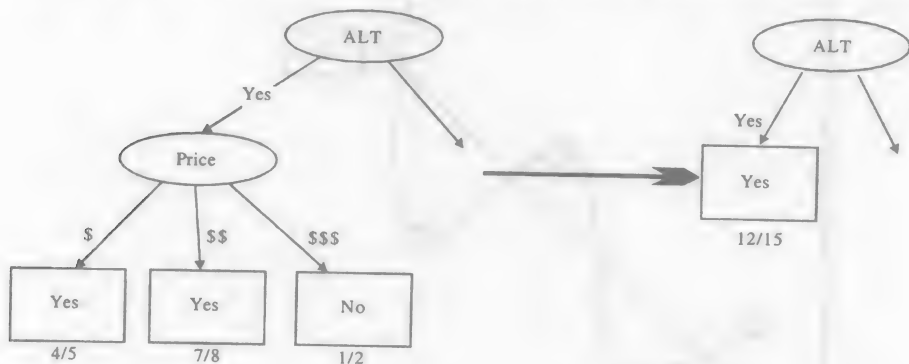


图4-20 子树置换

在进行子树置换时,从树叶向上到根进行处理。例如,在图4-21中,将考虑用一个叶节点置换“卫生计划贡献”子树的3个子女节点。假设决定进行这个置换,则继续沿树叶向上,考虑用一个叶节点置换“每周工作小时”子树(现在它有两个子女节点)。

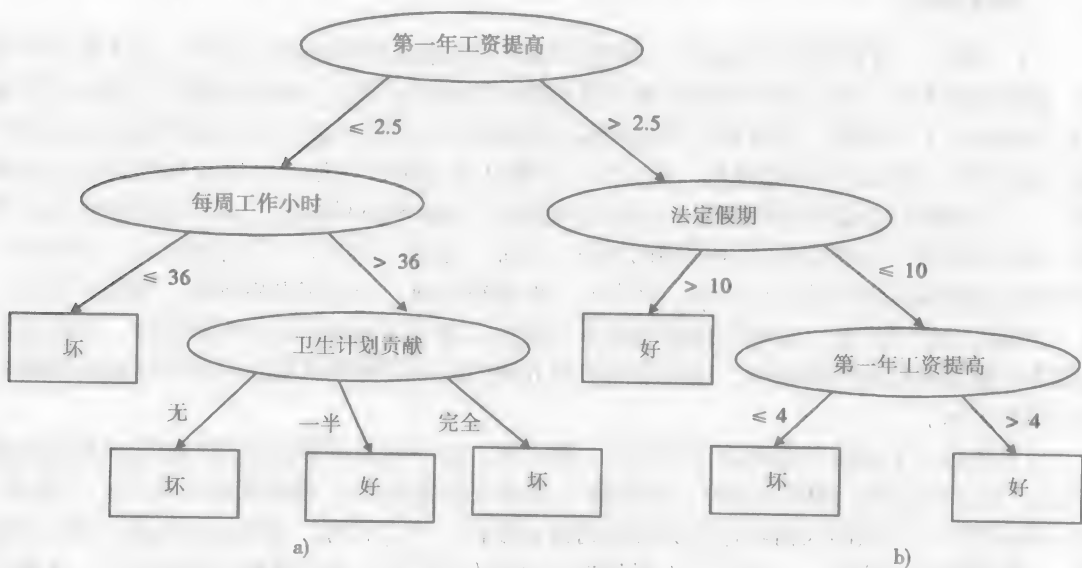


图4-21 一棵用于剪枝操作的树

子树提升:第二个剪枝操作“子树提升”更复杂。考虑图4-22。这里,考虑对图4-22a中的树剪枝,结果显示在图4-22b中。从C向下的整个子树被“提升”以置换B子树。注意,尽管B和C的子树显示为树叶,但是它们可以是整棵子树。当然,如果进行该提升操作,则必须考虑将标记为4和5的节点上的实例重新分类到以C开始的新子树中。这就是该节点的子女用1'、2'和3'标记的原因——指出它们与原来的子女1、2和3不同,差别在于包含了原来被4和5覆盖的实例。

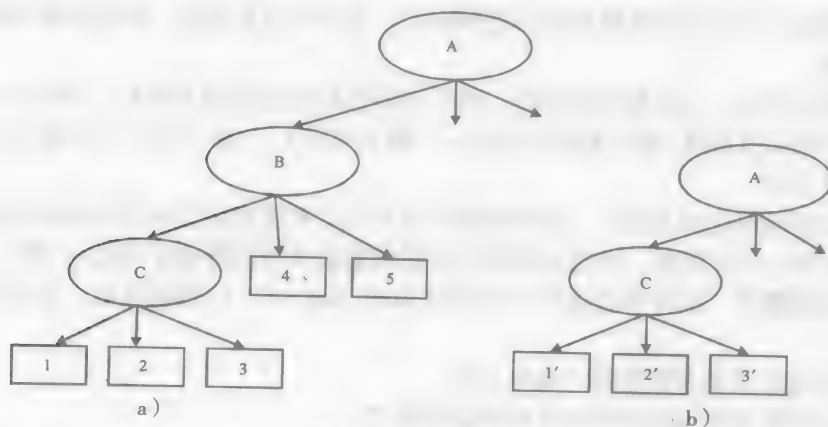


图 4-22 子树提升的例子，其中节点 C 被“提升”以包含节点 B

子树提升可能是一种耗时的操作。在实际实现时，一般只提升最普遍分支的子树。也就是说，在图 4-22 所示的例子中，倘若从 B 到 C 的分支的训练实例比从 B 到 4 或从 B 到 5 的分支多，我们就考虑进行提升。否则，如果 4 是 B 的多数子女，我们将考虑提升节点 4 来置换 B ，并将 C 下以及 5 的所有实例重新分类到新节点。

剪枝算法

图 4-23 的决策树使用 CART 算法学习得到。该树构建后在 60 个实例上检验。这个决策树也用来评审多种决策树剪枝方法。

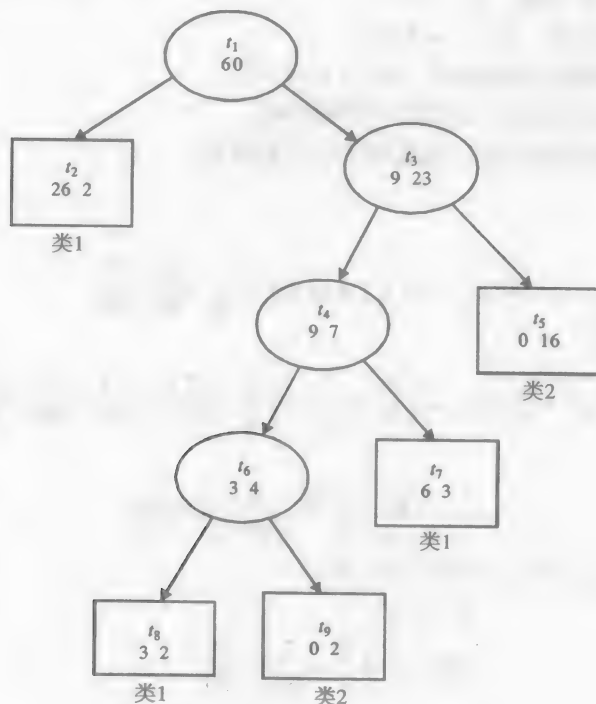


图 4-23 部分被剪枝的决策树的例子

决策树剪枝可以视为决策树构造的逻辑延续, 有两个重要原因: 降低决策树的规模和提高它的准确率。

有三种剪枝方法: 代价复杂性剪枝、最小误差剪枝和悲观误差剪枝。两种方法(代价复杂性剪枝和悲观误差剪枝)给出相同的建议——剪去节点 t_4 上的子树, 但是最小误差剪枝方法建议保留该子树。

在关于决策树剪枝的文献中, 达到的结果也不同: 有些方法产生更准确的决策树, 有些方法产生规模较小的决策树。结论是剪枝可以将归纳决策树的准确率提高达 25% [14]。

代价复杂性剪枝: 这种方法也称为 CART (Breiman 等) [1] 剪枝算法。它由两个基本步骤组成:

- 1) 根据某种启发式方法选择一系列子树。
- 2) 根据这些树的真误差率的估计选择最佳树 T_i 。

第一步的基本思想是通过剪去符合以下条件的分支从 T_i 得到 T_{i+1} : 这些分支中每个被剪树叶的表面误差率 (apparent error rate) 增加最小。

当树 T 在节点 t 被剪枝时, 它的表面误差率增加 $R(t) - R(T_i)$, 而树叶的数量减少 $|N_{T_i}| - 1$ 。这样, 下面的比率

$$\alpha = \frac{R(t) - R(T_i)}{|N_{T_i}| - 1} \quad (4.19)$$

度量每个被剪树叶的表面误差率的增加 [17]。算法对每棵子树 (第一棵除外) 计算 α , 并选择具有最小 α 值的子树进行剪枝。

为了解释这一点, 我们考虑图 4-23 中的节点 t_4 。现在

N_{T_i} = 子树 T_i 中的节点数, $N_{T_i} = 3$ 。

$r(t)$ = 节点 t 的误差率, $r(t_4) = 7/16$ 。

$p(t)$ = 子树 t 上数据所占的比例, $p(t_4) = 16/60$ 。

$R(t)$ = 节点 t 的误差代价, 如果该子树被剪枝。

$R(T_i)$ = 子树 T_i 的误差代价, 如果该节点不被剪枝。

i = 子树树叶。

于是

$$R(t_4) = r(t_4) \times p(t_4) = \frac{7}{16} \times \frac{16}{60} = \frac{7}{60}$$

并且

$$R(T_{i4}) = \sum R(i) = \left(\frac{2}{5} \times \frac{5}{60}\right) + \left(\frac{0}{2} \times \frac{2}{60}\right) + \left(\frac{3}{9} \times \frac{9}{60}\right) = \frac{5}{60}$$

这样,

$$\alpha = \frac{7/60 - 5/60}{3 - 1} = \frac{1}{60} \approx 0.0166$$

如果我们选择剪去节点 t_4 上的子树, 则

$$N_{T_{i6}} = 2$$

$$R(t_6) = \left(\frac{3}{7} \times \frac{7}{60}\right) = \frac{3}{60}$$

并且

$$R(T_{i6}) = \sum R(i) = \left(\frac{2}{5} \times \frac{5}{60}\right) + \left(\frac{0}{2} \times \frac{2}{60}\right) = \frac{2}{60}$$

这样,

$$\alpha = \frac{3/60 - 2/60}{2 - 1} = \frac{1}{60} \approx 0.0166$$

现在, 让我们看一看如果树在节点 t_3 剪枝, 每个被剪枝的树叶的表面误差率将如何增加。

$$N_{T_a} = 4$$

$$R(t_3) = \left(\frac{9}{32} \times \frac{32}{60} \right) = \frac{9}{60}$$

并且

$$R(T_{t_3}) = \sum R(i) = \left(\frac{2}{5} \times \frac{5}{60} \right) + \left(\frac{0}{2} \times \frac{2}{60} \right) + \left(\frac{3}{9} \times \frac{9}{16} \right) + \left(\frac{0}{16} \times \frac{16}{60} \right) = \frac{5}{60}$$

这样,

$$\alpha = \frac{9/60 - 5/60}{4 - 1} = 4/180 \approx 0.0222$$

最佳树是子树在节点 t_4 被剪枝后得到的树 (参见图 4-24), 因为其 α 值最小, 并且树的规模比在节点 t_6 剪枝得到的树小。

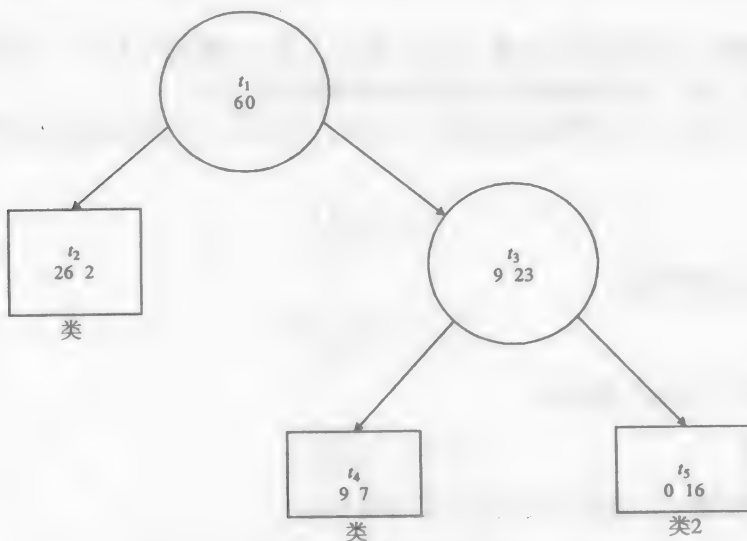


图 4-24 剪枝后的决策树

最小误差剪枝: Niblett 和 Bratko [18] 提出了一种方法, 寻找这样一棵树: 在对独立的数据集分类时, 使期望误差率最小。

设给定集合中的类数为 k , 节点 t 中训练实例的总数为 $n(t)$, 而节点 t 中属于类 C 的训练实例数为 $n_C(t)$, 则该期望误差率用如下公式计算:

$$E_k = \frac{n(t) - n_C(t) + k - 1}{n(t) + k} \quad (4.20)$$

剪枝方法如下所示。在树的每一个非叶节点, 计算子树被剪枝的期望误差率 E_k 。然后使用每个分支的误差率, 按照每个分支的观测值的比例加权, 计算节点不剪枝的期望误差率。如果剪去节点会产生较大的期望误差率, 则留下子树, 否则剪掉它 [16, 17]。

对于节点 t_6 , 期望误差率如下:

- 如果子树被剪枝

$$E_k = \frac{7-4+2-1}{7+2} = \frac{4}{9} \approx 0.4444$$

- 如果子树不剪枝

$$E_k = \frac{5}{7} \left(\frac{5-3+2-1}{5+2} \right) + \frac{2}{7} \left(\frac{2-2+2-1}{2+2} \right) \approx 0.3775$$

剪枝的期望误差更大, 因此决定不剪枝。

对于节点 t_4 , 期望误差率如下:

- 如果子树被剪枝

$$E_k = \frac{16-9+2-1}{16+2} = \frac{8}{18} \approx 0.4444$$

- 如果子树不剪枝

$$E_k = \frac{7}{16} * 0.3775 + \frac{9}{16} \left(\frac{9-6+2-1}{9+2} \right) \approx 0.3697$$

剪枝的期望误差还是较大, 因此决定不剪枝。这种方法的缺点之一是期望误差率依赖于类的个数。

悲观误差剪枝: 这种剪枝方法由 Quinlan 提出, 并且当前用于 C4.5, 目的是避免使用独立的检验数据集[16]。悲观剪枝基于误差数和训练样本的大小。

如果 $N(t)$ 是节点 t 上的训练集实例数, 而 $e(t)$ 是节点 t 上被误分类的实例数, 则误分类率的估计为:

$$r(t) = \frac{e(t)}{N(t)} \quad (4.21)$$

连续性校正误差率是:

$$r'(t) = \frac{e(t) + 1/2}{N(t)} \quad (4.22)$$

据此, 子树 T_t 的误分类率为:

$$r(T_t) = \frac{\sum e(i)}{\sum N(i)} \quad (4.23)$$

其中 i 遍取子树的树叶。这样, 校正后的误分类率是:

$$r'(T_t) = \frac{\sum (e(i) + 1/2)}{\sum N(i)} = \frac{\sum e(i) + N_{T_t}/2}{\sum N(i)} \quad (4.24)$$

其中 N_{T_t} 是节点 t 上的树叶数[16]。

使用训练数据, 子树总是比对应的节点产生的误差小, 但是使用校正后的数字时却并非如此, 因为它们依赖于树叶数, 而不仅仅是错误数。算法仅维持这样的子树: 其校正后的数字比节点的数字好一个标准差[14]。

标准差计算方法如下:

$$SE[n'(T_t)] = \sqrt{\frac{n'(T_t) * (N(t) - n'(T_t))}{N(t)}} \quad (4.25)$$

其中, 对于节点有

$$n'(t) = e(t) + 1/2$$

而对于子树有

$$n'(T_i) = \sum e(i) + N_{T_i}/2$$

因此, 如果子树的校正后的误分类个数大于节点的校正后的误分类个数, 该剪枝方法建议剪掉子树。

例如, 节点 t_4 上校正后的误分类数是:

$$n'(t_4) = 7 + 1/2 = 7.5$$

而子树的校正后的误分类个数是:

$$n'(T_{t_4}) = (2 + 0 + 3) + 3/2 = 6.5$$

$$SE = \sqrt{\frac{6.5 * (16 - 6.5)}{16}} \approx 1.96$$

由于 $6.5 + 1/96 = 8.46$, 大于 7.5 , 因此该子树将被剪枝。

这种方法有一些优点: 相同的训练集用于树生长和树剪枝, 并且它非常快, 因为只需要扫描一次并考察每个节点一次[16]。

临界值剪枝: 临界值剪枝方法由 Mingers 提出, 包括如下两个基本步骤[14]:

- 1) 关于增加的临界值对 T_{\max} 剪枝。
- 2) 通过度量整个树的显著性和它的预测能力, 在剪枝树序列中选择最佳树。

尽管这种方法是一种后剪枝方法, 但是它与先剪枝技术非常相似。在创建原始树时, 一种划分优良度量确定节点上的属性。这个度量值反映了所选择的属性的优良程度。

在实践中, 对于每个节点, 该方法计算子树的最大“信息增益”, 并且如果该值小于某个阈值就剪去一个节点[15]。

降低误差剪枝: 这是 Quinlan 提出的另一种方法, 它使用独立的样本检验每棵子树的准确率, 与它被剪枝时的准确率比较。该方法如下:

从整个树开始, 并对检验数据分类。对于每一个非叶节点, 统计保留该子树和通过剪枝将其变成树叶的误分类个数。在检验数据上, 被剪去的节点产生的错误通常比子树更少。错误个数之差用于度量树剪枝增益。对于所有的节点, 选择差最大的子树作为被剪枝子树[16]。这种方法的优点是每个节点只访问一次, 评估对它剪枝的机会。

4.8 模型评估

评估是使数据挖掘取得实际进展的关键。在数据挖掘过程的最后阶段, 使用一种或多种归纳学习技术得到模型之后, 仍然还存在一些重要问题:

- 1) 如何验证和确认模型?
- 2) 对于一个具体问题, 使用哪种方法?
- 3) 如何将一种方法与另一种比较?

首先, 我们来区分一下确认(validation)和验证(verification)这两个概念。

模型确认用合格检验证明模型在其应用范围内, 按照用户确定的目标, 以满意的正确率进行工作。换言之, 在模型确认中, 我们证实数据转换为模型, 并且它在表示被观测系统方面具有足够精度。模型确认处理构造正确的模型——对应于系统的模型。

模型验证证实模型是由数据转换来的、具有足够精度的新表示。模型验证处理正确地构造模型——对应于数据的模型。

数据挖掘结果通过检验过程加以确认和验证。某些检验用来评估模型的行为的正确性(即确认), 而另一些检验旨在评估数据转换成模型的正确性(即验证)。

我们有训练集, 当然可以在训练集上考察方法的不同之处。但是我们已经看到模型在训

训练集上的性能绝对不能很好地指示模型的性能,因为在非独立的检验集上,模型可能过分拟合数据,从而给出不准确的结果。我们需要基于能够得到的数据上的实验预测实际性能界限的方法。

在有丰富的数据可用时不存在问题:在一个大训练集上构造模型,并在另一个大检验集上检验它。但是,尽管数据挖掘有时涉及“大数据”(特别是在营销、销售和顾客支持应用中),但是通常数据(高质量的数据)是短缺的。

基于有限数据预测性能是一个有趣并且一直有争议的问题。

如果样本数量较小,那么数据挖掘实验的设计者就必须非常小心地划分数据。如何将样本划分成子集没有现成的指导原则。无论如何划分数据,都应当明白,不同的随机划分,即使训练集和检验集都具有给定的规模,也将导致不同的误差估计。

下面来讨论一下将数据集划分为训练和检验样本的不同方法,通常称作子抽样方法(resampling method)。使用子抽样方法估计和选择模型与使用分析方法相比的主要优点是,前者不依赖于关于数据统计分布的假定或逼近函数的特定性质。子抽样技术的主要缺点是计算量大且基于子抽样策略估计方差较高。

模型估计的基本方法是:首先使用一部分数据集准备或发现模型,然后使用其余样本评估该模型的预测风险。第一部分数据称为学习集(learning set),而第二部分数据称为确认集(validation set),也称为检验集(testing set)。这种朴素策略(naïve strategy)基于如下假定:学习集和确认集是作为相同的、未知的数据分布的代表而选取的。对于大型数据集的确如此,但是对于较小的数据集,这种策略具有明显的缺点。如果样本数较小,划分数据的具体方法对模型的准确率有所影响。各种子抽样方法用于较小的数据集,并且它们也因用于划分初始数据集的策略而异。我们将提供一些数据挖掘常用的子抽样方法的简单介绍,而数据挖掘系统的设计者必须根据数据和问题的性质进行选择。

再代入方法:这是最简单的方法。所有可用的数据都用于训练和检验。换句话说,训练和检验集相同。“数据分布”的误差率估计是偏向乐观的(估计的误差通常比模型实际应用期望的误差低),因此这种方法很少在现实世界的数据挖掘应用中使用。在样本大小与维度的比不大时尤其如此。

4.8.1 交叉确认:保持方法

现在考虑训练和检验数据量有限时该如何做。保持方法(holdout method)为检验保留一定数量的样本,并使用其余样本进行训练(如果需要的话,用一部分样本进行确认)。在实践中,通常为检验保留1/3数据,而使用其余2/3数据进行训练。

不同的划分将产生不同的估计。重复该过程,随机选择不同的训练和检验集,并将误差结果集成到一个标准参数中将改善模型的估计。这是误差率估计的重复保持(repeated hold-out)方法。

根据所使用的用于选择训练和检验集的抽样类型,基本有两种保持方法。抽样可以是有放回或无放回的。下面两种方法使用无放回抽样,而最后一种方法使用有放回抽样。

留一方法:模型使用 $(n-1)$ 个样本训练,而在剩下的一个样本上评估。这种方法重复 n 次,适用大小为 $(n-1)$ 的不同训练集。这种方法的计算量很大,因为必须构造和比较 n 个不同的模型。

轮转方法(n 折交叉确认):这种方法是保持和留一方法的折衷。它将可用的样本划分成 P 个不相交的子集,其中 $1 \leq P \leq n$ 。 $(P-1)$ 个子集用于训练,而剩下的一个子集用于检验。

这是实践中最常用的方法，特别是对样本数相对较小的问题尤其如此。

自助方法：前面说过，当一个样本从数据集中取出以形成训练或检验集时，样本被抽出而不放回。也就是说，一旦被选中，相同的样本不能被再次选中。这就像挑选足球队员，你不可能两次选择同一个人。但是，数据机实例与人不同。大部分学习方案可以两次使用相同的实例，并且如果实例在训练集出现两次，会使得学习结果不同。（拘泥数学的人会说，如果相同的对象可以多次出现，实际上根本不能说“集合”。）

自助方法的基本思想是对数据集进行有放回抽样，以形成训练集。我们将介绍一种变形，神秘地（但其理由很快就会清楚）称作 0.632 自助法 (0.632 bootstrap)。该方法对有 n 个实例的数据集有放回地抽样 n 次，产生另一个有 n 个实例的数据集。由于第二个数据集中的某些元素（几乎肯定）是重复的，因此原数据集中一定有一些实例未被选中。我们将使用这些实例作为检验实例。

一个实例未被选到训练集中的可能性有多大？实例每次被选中的概率是 $1/n$ ，因此不被选中的概率为 $(1 - 1/n)$ 。将它乘以挑选机会次数 n ，结果一个具体的实例完全未被选中的可能性为

$$\left(1 - \frac{1}{n}\right)^n \approx e^{-1} = 0.368$$

（其中 e 是自然对数的底 2.7183，不是误差率！）这样，对于大小合理的数据集，检验集将包含大约 36.8% 的实例，而训练集将包含大约 63.2%（现在你知道为什么称它为 0.632 自助法了）。有些实例将在训练集中重复出现，导致它的总规模为 n ，与原数据集的规模相同。

通过在训练集上训练学习系统，然后在检验集上计算它的误差，这样得到的数字将是真实误差率的悲观估计。因为尽管训练集的大小为 n ，但是只包含了 63% 的实例，这与 10 折交叉确认使用的 90% 的实例不是公平的比较。为了对此进行补偿，将该检验误差率与训练集上再代入误差相结合。正如我们前面告诫的，再代入误差是真实误差的过于乐观的估计，它自己不能用作误差估计。但是，自助过程将它与检验误差率相结合，给出最终的误差估计 e 如下：

$$e = 0.632e_{\text{检验实例}} + 0.368e_{\text{训练实例}}$$

然后，将整个自助过程重复多次，使用不同的替代样本作为训练集，并且对结果取平均值。

4.8.2 模型比较

通过数据挖掘过程使用不同的归纳学习技术实现的模型可以使用标准误差率参数作为其性能度量进行评估。这个值表示真实误差率的一种近似，一个统计学习理论定义参数。误差率使用通过再抽样技术得到的检验数据集计算。除用误差率度量的准确率之外，数据挖掘模型还可以用它们的速度、鲁棒性、可伸缩性和可解释性来比较。而且，所有这些参数都会影响模型的最终验证和确认。在下面的简略概述中，我们对分类任务解释误差率参数的特性，类似的方法和分析可以用于其他常见的数据挖掘任务。

误差率的计算基于检验过程的错误计数。对于分类问题，这些错误简单地定义为误分类（将样本错误分类）。如果所有的错误都同等重要，则误差率 R 是错误数 E 除以检验集中的样本数 S 。模型的正确率 A 是被正确分类的检验数据所占的比例，并且用 1 减误差率计算。

对于标准的分类问题，可能有多达 $m^2 - m$ 类错误，其中 m 是类的数目。如果只有两个

类(正样本和负样本,用符号 T 和 F 或用 1 和 0 表示),我们只可能有两类错误:

- 预期为 T , 但错误地分类为 F : 这些是假阴性错误。
- 预期为 F , 但错误地分类为 T : 这些是假阳性错误。

如果多于两个类,错误类型可以用表 4-18 所示的混淆矩阵汇总。对于类数 $m=3$, 有 6 个错误类型($m^2 - m = 3^2 - 3 = 6$), 它们在表 4-18 中用粗体字表示。在这个例子中, 每个类包含 30 个样本, 共有 90 个检验样本。

这个例子的误差率是 10/90, 总对应的正确率为 80/90。

至此, 我们考虑每个错误都一样不好。在许多数据挖掘应用中, 假定所有的错误都具有相同的权重是不能接受的。因此, 应当记录各种错误的差别, 并且误差率的最终度量要考虑这些差别。当不同类型的错误与不同的权重关联时, 我们需要用给定的权重因子 c_{ii} 乘以误差每种类型。如果混淆矩阵的误差元素是 e_{ii} , 则总代价函数 C (它取代准确率计算中的错误数) 可以用下式计算:

$$\sum c_{ii} e_{ii}$$

表 4-18 3 个类的混淆矩阵

分类模型	真实的类			总数
	0	1	2	
0	28	1	4	33
1	2	28	2	32
2	0	1	24	25
总数	30	30	30	90

4.8.3 代价敏感的学习

在两类情况下, 有一种简单而有效的方法使得任意学习方案都是代价敏感的。其基本思想是通过产生 yes 和 no 实例所占比例不同的数据样本, 使得学习方法是代价敏感的。假设人为地将数据集中 no 实例增加 10 倍, 然后使用该数据集进行训练。如果学习方案力求使错误数最小化, 则它将产生一个决策结构, 朝着避免 no 实例上的错误倾斜, 因为这种错误实际上被 10 倍地加以处罚。如果使用 no 实例所占比例未变的数据进行检验, 则在 no 实例上的错误将比在 yes 实例上的错误少(即假阳性错误将比假阴性错误少), 因为假阳性的权重是假阴性的 10 倍。于是产生了一种通过改变训练集中实例的比例建立代价敏感分类结构的一般技术。

改变训练实例比例的一种方法是复制数据集中的实例。然而, 许多学习方案支持实例被加权的。缺失值可以用以下方法提供: 在建立决策树时通过使用数值加权方案, 在概念上将实例划分成片段, 并将它的某个部分传送到每个分支。

实例的权重通常初始化为 1。为了构造代价敏感的决策树, 可以将它们初始化为两类错误(假阳性和假阴性)的相对代价。

习题

1. 判断对错。

1) 如果 $P(A|B) = P(A)$, 则 $P(A \cap B) = P(A)P(B)$ 。

2) 因为决策树学习对离散值输出分类, 而不是对实数值函数分类, 因此它们不可能过分

拟合。

2. 概率分布[0:0625; 0:0625; 0:125; 0:25; 0:5]的熵是多少?
3. 汽车保险例子。假定训练数据库具有两个属性: 年龄和汽车类型。
 - 年龄——序数属性。
 - 汽车类型——分类属性。
 - 类——L: 低(风险), H: 高(风险)。

年龄	汽车类型	类
>21	Maruti	L
>21	Hyundai	H
<21	Maruti	H
<21	Indica	H
>21	Maruti	L
>21	Hyundai	H

使用 ID3 算法得到一棵决策树。

4. 你被搁浅在一个荒岛上。岛上到处都长满了蘑菇, 但是找不到其他食物。有些蘑菇已被确定是有毒的, 而其他无毒(通过先前同伴的试验和错误而确定)。你是唯一留在荒岛上的人。你有如下数据:

实例	厚实否	有味否	有斑点否	光滑否	有毒否
A	0	0	0	0	0
B	0	0	1	0	0
C	1	1	0	1	0
D	1	0	0	1	1
E	0	1	1	0	1
F	0	0	1	1	1
G	0	0	0	1	1
H	1	1	0	0	1
U	1	1	1	1	?
V	0	1	0	1	?
W	1	1	0	0	?

你知道蘑菇 A ~ H 是否有毒, 但不知道 U ~ W 是否有毒。对于前两个问题, 只考虑 A ~ H。

- 1) “有毒否”的熵是多少?
- 2) 你应当选择哪个属性作为决策树根节点? (提示: 你可以通过观察数据断定, 而不必计算所有 4 个属性的信息增益。)
- 3) 使用 ID3 算法构造一棵决策树, 并预测案例 U、V 和 W。
- 4) 使用 CART 算法构造一棵决策树, 并预测案例 U、V 和 W。
- 5) 使用 CHAID 算法构造一棵决策树, 并预测案例 U、V 和 W。
5. 假定你是所得税部门的高级官员。你得到了应当纳税的纳税人以往记录的代表样本。下面的表显示了这些数据。导出一棵决策树提醒执法部门。(取划分点为 800K, 将“纳税收入”转换成二元变量。)

Tid	政府雇员	婚姻状况	纳税收入	逃税
1	是	单身	1250K	否
2	否	已婚	1000K	否
3	否	单身	700K	否
4	是	已婚	1200K	否
5	否	离婚	950K	是
6	否	已婚	600K	否
7	是	离婚	2200K	否
8	否	单身	950K	是
9	否	已婚	750K	否
10	否	单身	90K	否

6. 假定你是学院的篮球队队长。根据下表给出的记录，设计一种赢得下场比赛的策略。

地点	时间	Sachin 首发	Girish 进攻	Girish 防御	对手中锋	结果
学院	7pm	是	中锋	前锋	高矮	赢
学院	7pm	是	前锋	中锋	矮	赢
大学	7pm	是	前锋	前锋	高矮	赢
大学	9pm	是	前锋	前锋	矮	输
学院	7pm	是	中锋	中锋	高矮	赢
大学	7pm	是	中锋	中锋	矮	赢
大学	9pm	是	中锋	前锋	矮	输
学院	7pm	是	中锋	中锋	矮	赢
学院	7pm	是	中锋	前锋	矮	赢
学院	7pm	是	中锋	前锋	高	赢

7. 对下表所示数据，使用 ID3、CART 和 CHAID 算法构造决策树。

x_1	x_2	x_3	y	x_1	x_2	x_3	y
低	低	低	低	中	中	高	中
低	低	高	低	中	高	低	中
低	中	低	中	中	高	高	高
低	中	高	低	中	低	低	高
低	高	低	高	高	低	高	高
低	高	高	中	高	中	低	高
中	低	低	中	高	中	高	高
中	低	高	中	高	高	低	高
中	中	低	中	高	高	高	高

8. 对下表所示数据，使用 ID3、CART 和 CHAID 算法构造决策树。

名称	体温	表皮覆盖	胎生	产蛋	能飞	水生	有腿	冬眠	类标记
人	恒温	毛发	是	否	否	否	是	否	哺乳类
巨蟒	冷血	鳞片	否	是	否	否	否	是	爬行类
鲑鱼	冷血	鳞片	否	是	否	是	否	否	鱼类
鲸	恒温	毛发	是	否	否	是	否	否	哺乳类
蛙	冷血	无	否	是	否	有时	是	是	两栖类
巨蜥	冷血	鳞片	否	是	否	否	是	否	爬行类
蝙蝠	恒温	毛发	是	否	是	否	是	是	哺乳类
鸽子	恒温	羽毛	否	是	是	否	是	否	鸟类

(续)

名称	体温	表皮覆盖	胎生	产蛋	能飞	水生	有腿	冬眠	类标记
猫	恒温	皮	是	否	否	否	是	否	哺乳类
豹纹鲨	冷血	鳞片	是	否	否	是	否	否	鱼类
海龟	冷血	鳞片	否	是	否	有时	是	否	爬行类
企鹅	恒温	羽毛	否	是	否	有时	是	否	鸟类
豪猪	恒温	刚毛	是	否	否	否	是	是	哺乳类
鳗	冷血	鳞片	否	是	否	是	否	否	鱼类
蝾螈	冷血	无	否	是	否	有时	是	是	两栖类

9. 下图给出各种飞机的图像, 我们希望建立一个系统, 可以将飞机分成国产和进口两类。为了简化该问题, 我们假定不处理图像理解过程。一种计算机视觉预处理程序考察飞机图像, 并提取我们分类时必须考察的相关特征。在这个例子中, 我们严格关注使得计算机可以进行这种分类的知识的获得。该过程从识别感兴趣的飞机属性以及它们的可能值开始。这在表 4-19 给出。知识工程师或专家必须开发用来导出规则的实例集。在我们的例子中, 图像所具有的相关特征和它的分类(国产或进口)描述每幅图像。这些在表 4-20 中。实例矩阵称作归纳文件(induction file)。

使用 ID3 算法产生最小树, 从而是最简单的可能规则。

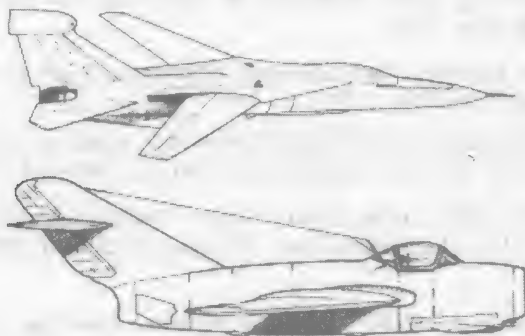


表 4-19 属性和可能的值

特 征	可能的值
机翼安装	高, 中, 低
发动机	1~3
机头	平, 短平, 突出
进风口	机头, 机身
机身	雪茄形, 圆滑, 粗大

表 4-20 归纳表

机翼安装	发动机	机头	进风口	机身	类别
中	1	平	机头	雪茄形	进口
中	1	平	机头	圆滑	进口
低	1	短平	机头	圆滑	进口
高	2	突出	机身	粗大	国产
高	1	突出	机身	粗大	国产

参考文献

- [1] L. Breiman, J.H. Friedman, R.A. Olshen and C.J. Stone, *Classification and Regression Trees*, 1984.
- [2] J. Quinlan, *C4.5 Programs for Machine Learning*, Morgan Kaufmann Publishers, 1993.
- [3] G.V. Kass, An exploratory technique for investigating large quantities of categorical data, *Applied Statistics*, 29, pp 119–127, 1980.
- [4] D.M. Hawkins and G.V. Kass, Automatic interaction detection. In: D.M. Hawkins (Ed.), *Topics in Applied Multivariate Analysis*, Cambridge University Press, 1982.
- [5] N. Drapper and H. Smith, *Applied Regression Analysis*, 2nd ed., Wiley Series in Probability and Mathematical Statistics, 1981.
- [6] D. Harrison and D. Rubinfeld, Hedonic prices and the demand for clean air, *Journal of Environment Econ. and Management*, 5, pp. 81–102, 1978.
- [7] J. Morgan and J. Sonquist, Problems in the analysis of survey data, and a proposal, *Journal of American Statistics Society*, 58, pp. 415–434, 1963.
- [8] T. Hastie and R. Tibshirani, *Generalized Additive Models*, Chapman & Hall, London 1990.
- [9] A. Karalic, Employing linear regression in regression tree leaves. In: *Proceedings of ECAI-92*, Wiley & Sons, 1992.
- [10] W.D. Fisher, On grouping for maximum homogeneity, *Journal of American Statistical Association*, 53, pp. 789–798, 1958.
- [11] P.A. Chou, Optimal partitioning of classification and regression trees, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4), pp. 340–354, 1991.
- [12] H. Ian Witten and Eibe Frank, *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*, Morgan Kaufmann Publishers, San Francisco, CA, 2000.
- [13] C. Schaffer, Overfitting avoidance as bias, *Machine Learning*, 10(2), Kluwer Academic Publishers, 1993.
- [14] J. Mingers, An empirical comparison of pruning methods for decision tree induction, *Machine Learning*, 4, pp. 227–243, 1989.
- [15] Y. Mansour, Pessimistic decision tree pruning based on tree size, *Proc. 14th International Conference on Machine Learning*, 1997.
- [16] J.P. Ignizio, *An Introduction to Expert Systems: The Development and Implementation of Rule Based Expert Systems*, McGraw Hill, Inc., p. 402, 1991.
- [17] F. Esposito, D. Malerba, and G. Semeraro, A comparative analysis of methods for pruning decision trees, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 5, pp. 476–491, 1997.
- [18] T. Niblett and I. Bratko, Learning decision rules in noisy domains, In: M.A. Bramer, *Research and Development in Expert Systems III*, Cambridge University Press, Cambridge, pp. 25–34, 1986.

第5章 数据挖掘的预处理和后处理

5.1 引言

数据挖掘过程最重要的步骤之一是初始数据集的准备和变换。这项任务在研究界不太受重视,主要是因为该步骤过多地考虑具体应用。但是,在大部分数据挖掘应用中,有时数据准备过程的某些部分,甚至整个过程都可以独立地看作数据挖掘方法的应用。对于一些数据集非常大,并且通常分布在各处的公司,一些数据准备任务可以在设计数据仓库时进行,但是许多具体的数据变换只能在需要数据挖掘分析时才能进行。

原始数据很难用于数据挖掘。可能需要对其做一些变换,产生对诸如预测或分类这样的对选择数据挖掘方法更有用的特征。在大部分情况下,找到给定方法/应用的最佳变换需要人的协助。

在数据挖掘文献中,数据准备有时被看作小问题而不予考虑,而是把它作为数据挖掘过程的一个阶段。在实际的数据挖掘应用中,情况正好相反。用在数据准备上的工作量比用在使用数据挖掘方法上的工作量还多。在5.2节中,我们将简略介绍不同的预处理任务,它们在对数据使用数据挖掘算法之前进行。

5.2 数据预处理的步骤

选择对象表示:知识发现过程的输入是数据库,即对象的集合。作为给定问题的单位,数据对象必须用一组基本描述形式化地描述。因此,我们必须选择适当的对象表示。最常用的选择是对象的属性表示。称作属性(attribute)的基本性质用来描述实际对象。一个对象用属性和值的列表描述。每个属性具有一个可能的值域。根据属性域的组织,我们可以区分三种基本属性类型:符号的(离散的、标称的、分类的)、连续的(数值的)和结构化的。

映射和收集数据:选择合适的表示后,我们选择度量对象的属性(根据领域专家的建议或使用“蛮力”方法)。此外,我们还必须确定属性的名称和它们的值的名称。收集的数据映射到某个命名约定上并用一致的形式表示。

缩放大型数据集:实践中,学习算法假定数据都位于主存中,并且在只能看到有限个数据时,并不关心算法如何处理非常大的数据库。解决该问题的可能方法有多种,例如使用窗口、批增量模式。

处理噪声和错误:通常有两种类型的错误。外部(external)错误是从系统之外的世界引进的(随机错误和噪声)。内部(internal)错误是学习(数据挖掘)系统本身的不好的性质导致的。例如,启发式搜索或性能标准不佳。

处理未知属性值:在处理现实世界的数据时,一个特别重要的问题是处理未知的(缺失的)属性值。在处理未知的属性值时,需要考虑一些重要因素。最重要的因素之一是“未知性”的源:①一个值缺失是因为它被遗忘或丢失。②对于给定的对象,某些属性是不适当的。例如,对于给定的对象,它不存在。③在给定的背景下,属性值是不相关的。④对于给定的观测值,训练数据库的设计者并不关心某属性的值(因此称作不关心的值)。

数值属性的离散化或模糊化：有些学习算法只能处理符号或分类数据。然而，现实世界的问题既涉及符号属性，也涉及数值属性。因此，一个重要问题是将数值(连续)属性离散化。这一任务可以离线(预处理)进行，也可以在线(动态离散化)进行。该过程的一种自然扩展是数值属性的模糊化。

处理连续类：与上面的问题类似的问题是连续类的处理。大部分符号归纳算法要求离散的(符号)类。然而，在大部分应用中，我们面对的是连续(数值)类。类似于上面的问题，可以使用两种方法：离线或在线划分。

符号属性值分组：一个众所周知的问题是，对于归纳决策树和推导决策规则，具有很多值的属性在选择最富有信息的属性的过程中将被过高估计。为了克服过高估计问题，可以将多值属性的值分成两个或多个子集。

属性选择和定序：对于给定的目标，我们不能确保所有的属性都提供信息。在现实世界的的数据中，数据的表示通常使用了过多属性，但其中只有少量属性可能与目标概念相关。属性选择和定序过程将有助于解决这一问题。它们根据属性提供信息的多少确定输入属性集的顺序，然后选择提供信息最多但规模相对较小的属性子集。

属性构造和变换：找出问题表示的合适属性是一项耗时和困难的任务。对于目标概念，如果属性不合适，那么数据挖掘(学习)可能是困难的或者是不可能的。为了克服这一问题，系统要能够产生(构造)新的合适的属性。完成这件事有两种不同的方法：属性构造和属性变换。

一致性检查：对于数据库中的不一致性，它们也许不能被前面的预处理步骤消除。有两种处理数据不一致性的一般方法。第一种是“离线”方法，即通过预处理程序或在数据挖掘过程本身中处理。另一种可能的方法是利用知识发现过程的循环机制，即返回到前面的某个步骤，并对不同的参数重新执行。

现在，我们详细介绍这些预处理步骤中的某些步骤。

5.3 离散化

很多机器学习和统计学技术都只能用于完全由标称变量组成的数据集。然而，实际中的许多数据集包含连续变量——在区间或比率级测量的变量。解决该问题的一种方法是将数值变量的值域划分成一些子域，并将每个子域看作一个类别。这种将连续变量划分成不同类别的过程通常称为离散化(discretization)。

近年来，已经开发了各种离散化方法。Dougherty、Kohavi 和 Sahami [1] 给出了这方面工作的系统总结，其中对离散化技术从两个方面进行了说明：监督的(supervised)与非监督的(unsupervised)，全局的(global)与局部的(local)。

非监督的离散化过程只使用变量值的分布信息划分变量。相比之下，监督的过程还使用每个实例的分类标号。典型的非监督技术包括：

- 1) 等区间宽度方法，其中值域简单地化分成等宽的子域。
- 2) 等频方法，其中值域被划分成包含相同数目实例的子域。
- 3) 更复杂的非监督方法使用聚类分析技术，识别最大化组内相似性、最小化组间相似性的划分。

通常，监督技术试图使划分变量与类标号之间联系的某种度量最大化。方法包括：

- 1) 度量联系强度的熵或信息增益[2]。
- 2) χ^2 检验确定哪些组应当合并，ChiMerge[3]算法使用这种方法。

监督技术被理所当然地认为能产生更准确的分类树，因为它们产生的划分直接与所预测的类相关。另一方面，人们可以预料大部分非监督技术的速度明显较快，因为除了对数据排序之外，它们不太涉及其他操作，而排序是所有离散化方法的共同操作。

全局离散化过程在构建决策树过程开始之前作用于整个数据集。这样，当给定的变量在树中出现时，它将在相同的点划分。与之相反，局部离散化过程在树构造时被应用到与树节点相关联的实例子集。这样，随着树的构建，相同的变量可能离散化多次，最终的树可能包含相同变量的多种划分。

由于局部离散化技术可以对样本空间的不同部分产生不同的划分，可以预料它们在产生准确的分类树方面优于全局方法。然而，也可以预料它们为提高准确性付出的速度代价是不可忽略的，因为离散化过程将在树构造时重复多次。

在本章中，我们介绍在一些流行的数据挖掘算法中使用的方法。

5.3.1 人工方法

将连续特征值离散化成少量区间是特征离散化过程的任务，其中每个区间被映射为离散符号。在这种情况下，使用关于特征的先验知识。例如，个人薪水是连续值，在 1000 和 10 万卢比之间，在数据挖掘过程开始时，可以将其分类为：很低、低、中等、高和很高。这里，截断点是很主观地确定的(参见图 5-1)。

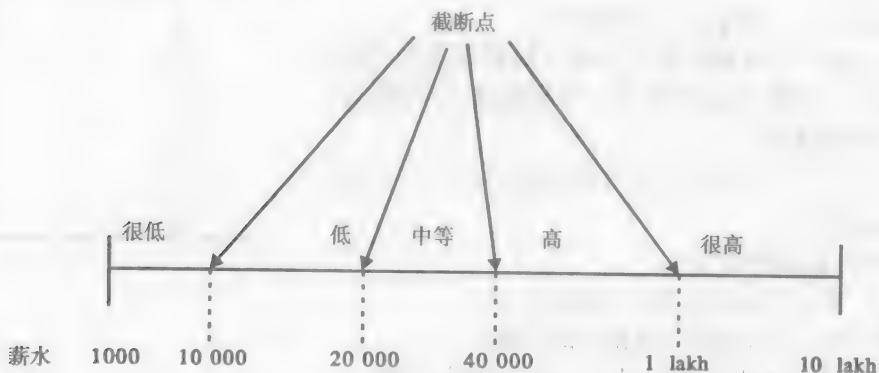


图 5-1 薪水变量的离散化

这个归约过程的两个主要问题是：

- 1) 什么是截断点？
- 2) 如何选择区间的表示？

如果没有关于特征的知识，离散化会困难得多，并且在许多情况下是非常任意的。通常，对于实际的数据挖掘应用，特征值的规约是无害的，并且会使计算复杂性降低。因此，在下面两节我们将介绍一些自动的离散化技术。

5.3.2 分箱

分箱用于每个特征(属性)。它不使用类信息。假定我们有属性“年龄”的如下值集合：0, 4, 12, 16, 16, 18, 24, 26, 28。有两种应用于分箱的可能方式：等宽分箱和等频分箱。

等宽分箱：让我们考虑箱宽度为 10 的情况。箱元素和箱边界如表 5-1 所示。

等频分箱：我们考虑箱密度为 3 的情况。现在，箱元素和箱边界显示在表 5-2 中。

表 5-1 等宽分箱

箱号	箱元素	箱边界
1	{0, 4}	[-, 10)
2	{12, 16, 16, 18}	[10, 20)
3	{24, 26, 28}	[20, +)

表 5-2 等频分箱

箱号	箱元素	箱边界
1	{0, 4, 12}	[-, 14)
2	{16, 16, 18}	[14, 21)
3	{24, 26, 28}	[21, +)

现在, 可以用箱均值、箱中值或箱边界置换箱中的每个值来离散化值集合。

5.3.3 基于熵的离散化

基于熵的方法使用数据中提供的类信息。熵(或信息内容)基于类标号来计算。下面是这种方法的一个例子, 称为通过二元化的**基于熵的分箱**(entropy-based binning)。直观地, 该方法找出最佳划分, 使得箱尽可能纯, 即箱中的大多数值对应于具有相同类标号的实例。形式地说, 它通过找出最大化信息增益的划分来进行离散化。

假设我们有如下(属性值/类)对。设 S 是表 5-3 给定的 9 个对。注意, 在离散化之前, 数据已经排序。

令 $p = 4/9$ 为类值等于 P 的对所占的比例, 而 $n = 5/9$ 为类值等于 N 的对所占的比例。 S 的熵(信息内容)由下式得到:

$$\text{Entropy}(S) = -p \log(p) - n \log(n)$$

设 $X = v$ 是一个可能的划分, 将 S 划分成两个子集 S_1 和 S_2 , 其中 S_1 是值 $X \leq v$ 的集合, S_2 是值 $X > v$ 的集合。

该划分的信息为

$$\text{Info}(S_1, S_2) = (|S_1|/|S|) \text{Entropy}(S_1) + (|S_2|/|S|) \text{Entropy}(S_2)$$

该划分的信息增益为:

$$\text{Gain}(v, S) = \text{Entropy}(S) - \text{Info}(S_1, S_2)$$

其中 $|S|$ 表示集合 S 的基数(数据点的个数)。

例如, 如果我们在属性值 $X = 14$ 上划分, 则

$$S_1 = \{(0, P), (4, P), (12, P)\}, S_2 = \{(16, N), (18, P), (24, N), (26, N), (28, N)\}$$

$$\text{Info}(S_1, S_2) = (3/9) \text{Entropy}(S_1) + (6/9) \text{Entropy}(S_2) = 0 + (6/9)0.9163 = 0.1308$$

$$\text{Gain}(14, S) = \text{Entropy}(S) - 0.1308$$

该算法的目标是找到具有最大信息增益的划分。当 $\text{Info}(S_1, S_2)$ 最小时, 信息增益最大。通过考察所有的划分, 选择最优划分得到最佳划分。

在实践中, 不必考虑所有可能的割点。Fayyad 和 Irani[4]证明使熵最小化的最佳割点一定在不同的类实例之间。对于我们考察的数据, 在割点 14、17 和 21 的熵计算在表 5-4 给出。由于最小熵在割点 14(属性值 12 和 16 之间)出现, 因此它被取做最佳割点。

表 5-4 基于熵的割点

属性值	0	4	12	16	16	18	24	26	28
类标号	P	P	P	N	N	P	N	N	N
在割点上的熵	$\text{Info}([3, 0], [5, 1]) = 0.433$			$\text{Info}([3, 2], [1, 3]) = 0.9$		$\text{Info}([4, 2], [0, 3]) = 0.6121$			

表 5-3 待离散化的数据

预测变量 X	目标变量 Y
0	P
4	P
12	P
16	N
16	N
18	P
24	N
26	N
28	N

5.3.4 找出分割点的其他简单方法

在本节中,我们介绍两种确定分割点的简单方法[5],它们的时间和空间复杂性随着相关记录的个数呈线性增长。所有三种方法都使用相同的离散化方法,称为分位数方法(quantile approach)。使用该方法,我们对决策树的每个节点上的分割点值假定一个分布,并选择分割点,使得该分布被划分成两个等概率区间。

高斯近似:第一种方法称为高斯近似(Gaussian approximation)方法,只需要知道与树中每个叶节点 L 相关联的数据记录的每个需要划分的连续预测子的均值和标准差。这些统计量容易在数据向下传递到新形成的划分的子女时收集到。对于向下传递到叶节点 L 的每个记录,更新 L 中每个连续预测子的累计和、累计平方和。当所有的数据到达 L 时,从这些和导出均值和标准差。

我们按以下方法得到叶节点 L 上每个连续预测子 X_j 的划分点。首先,我们选取被考虑的划分点数目 k 。这种选择可以动态地进行(即通过模型选择进行),或者在学习算法运行前,对所有节点的所有预测子预先定义 k 。第二,我们假设(通常不太好) X_j 在与 L 相关的实例上正态分布,选择这样的划分点,它们产生 X_j 的值域的 $k+1$ 个等密度区域。特殊地,设 $\{c_1, c_2, \dots, c_k\}$ 为 k 个划分点的集合,我们选择 c_i :

$$c_i = \mu_L + \sigma_L \cdot \Phi^{-1}\left(\frac{i}{k+1}\right)$$

其中, Φ^{-1} 是标准高斯累积分布函数的逆, μ_L 和 σ_L 分别是与 L 相关的 X_j 的值的均值和标准差。

k-分位数方法:在第二种计算划分点的方法中,我们使用 k -分位数选择连续划分点。这对应于使用经验分布函数的分位数方法。也就是说,对于连续预测子 X_j ,我们选择划分点,使得(大约)有 $m_L \frac{i}{k+1}$ 个记录满足 $X_j < C_i$, $m_L \frac{k-i}{k+1}$ 个记录满足 $X_j > C_i$ 。例如,如果 $k=1$,则该方法只选择与 L 相关的记录中 X_j 的中值。与前面的方法一样,我们可以对所有的节点和所有的预测子预先计算 k 。

ChiMerge

ChiMerge[6]是一种自动离散算法,它使用 χ^2 统计量分析给定特征的多个区间的质量。该算法根据样本的输出分类确定两个相邻区间中数据分布的相似性。如果 χ^2 检验的结论是类输出独立于特征区间,则区间应当合并;否则,它表明区间之间的差别是统计显著的,因此不进行合并。

对于离散化,ChiMerge 算法包括三个基本步骤:

- 1) 给定特征的数据按递增次序排序。
- 2) 确定初始区间,使得每个特征值在一个单独的区间中。
- 3) 重复合并相邻的区间,直到没有两个相邻区间的 χ^2 值小于阈值。

每次合并之后,计算剩余区间的 χ^2 值,并找出具有最小 χ^2 值的两个相邻特征。如果该 χ^2 值小于阈值,则合并这些区间。如果不能合并,并且区间数大于用户定义的最大值,则增加该阈值。

χ^2 检验或相依表检验用于确定两个相邻区间的独立性。当数据汇总在相依表中时(它的形式见表 5-5), χ^2 检验由下式给出:

$$\chi^2 = \sum_{i=1}^2 \sum_{j=1}^k \frac{(A_{ij} - E_{ij})^2}{E_{ij}}$$

其中, k 是类数, A_{ij} 是第 i 个区间中第 j 个类的实例数。 E_{ij} 是 A_{ij} 的期望频率, 用 $(R_i \cdot C_j)/N$ 计算。 R_i 是第 i 个区间中的实例数 $\sum_{j=1}^k A_{ij}$, C_j 是第 j 个类的实例数 $\sum_{i=1}^2 A_{ij}$, 而 N 是实例总数 $\sum_{i=1}^2 R_i$ 。

表 5-5 2×2 分类数据的相依表

	类 1	类 2	Σ
区间 1	A_{11}	A_{12}	R_1
区间 2	A_{21}	A_{22}	R_2
Σ	C_1	C_2	N

如果相依表中 R_i 或 C_j 为 0, 则设置 E_{ij} 为一个小值, 如 $E_{ij} = 0.1$ 。进行这一修正的理由是避免检验的分母出现非常小的值。对于给定的数据集, χ^2 检验的自由度是小于类个数的数。当离散化的特征多于一个时, 应当分别指定每个特征的区间最大个数阈值和 χ^2 检验置信区间。如果区间个数超过该最大值, ChiMerge 算法可能以一个新的、约减的置信值继续进行。

对于两个类的分类问题 ($k=2$), 分析两个区间的合并, 2×2 数据的相依表具有表 5-5 的形式。 A_{11} 表示第一个区间中属于第一个类的样本数, A_{12} 是第一个区间中属于第二个类的样本数, A_{21} 是第二个区间中属于第一个类的样本数, A_{22} 是第二个区间中属于第二个类的样本数。

我们将使用一个相对简单的例子分析 ChiMerge 算法, 其中数据库包含 12 个二维样本, 只有一个连续特征 (F) 和一个分类输出特征 (K)。特征 K 的两个值 1 和 2 表示样本所属的两个类。初始数据集如表 5-6 所示, 已经按照连续数之特征 F 进行了排序。

表 5-6 在连续特征 F 上排序具有对应类 K 的数据

样本	F	K	样本	F	K
1	1	1	7	23	2
2	3	2	8	37	1
3	7	1	9	39	2
4	8	1	10	45	1
5	9	1	11	46	1
6	11	2	12	59	1

下面开始离散化算法, 对 F 的排序数值区间选择最小的 χ^2 值。我们定义给定数据的中间值为划分区间点。例如, 对于我们的例子, 特征 F 的区间点为 0, 2, 5, 7.5, 8.5, 10 等。根据区间的这种分布, 我们分析所有相邻区间, 试图找出 χ^2 检验值最小的相邻区间。在我们的例子中, χ^2 检验值最小的相邻区间为 [7.5, 8.5] 和 [8.5, 10]。两个区间都只包含一个样本, 并且它们都属于类 $k=1$ 。初始相依表如表 5-7 所示。

表 5-7 区间 [7.5, 8.5] 和 [8.5, 10] 的相依表

	$k=1$	$k=2$	Σ
区间 [7.5, 8.5]	$A_{11} = 1$	$A_{12} = 0$	$R_1 = 1$
区间 [8.5, 10]	$A_{21} = 1$	$A_{22} = 0$	$R_2 = 1$
Σ	$C_1 = 2$	$C_2 = 0$	$N = 2$

根据表 5-7 给定的值, 我们可以计算期望值:

- $E_{11} = 2/2 = 1$
- $E_{12} = 0/2 \approx 0.1$ (期望值为 0, 通常选择一个较小的量)
- $E_{21} = 2/2 = 1$
- $E_{22} = 0/2 \approx 0.1$ (期望值为 0, 通常选择一个较小的量), 而对应的 χ^2 是:

$$\chi^2_{\text{计算的}} = (1-1)^2/1 + (0-0.1)^2/0.1 + (1-1)^2/1 + (0-0.1)^2/0.1 = 0.2$$

$$\chi^2_{\text{查表的}} = 2.706 \text{ (对于自由度 } d=1, \alpha=0.1, \text{ 阈值由卡方分布表得到)}$$

由于 $\chi^2_{\text{计算的}} < \chi^2_{\text{查表的}}$, 我们断言相对类频率, 所选择的区间没有显著差别, 可以合并。在每次迭代中, 只对具有最小 χ^2 值, 并且 χ^2 值小于阈值的两个相邻区间应用合并过程。对下两个具有最小 χ^2 值的相邻区间继续该迭代过程。我们只给出合并过程中间的某处的一个附加步骤, 其中分析了区间 $[0, 7.5]$ 和 $[7.5, 10]$ 。相依表在表 5-8a 给出, 而期望值为:

- $E_{11} = 12/5 = 2.4$
- $E_{12} = 3/5 = 0.6$
- $E_{21} = 8/5 = 1.6$
- $E_{22} = 2/5 = 0.4$

而 χ^2 检验为:

$$\chi^2_{\text{计算的}} = (2-2.4)^2/2.4 + (1-0.6)^2/0.6 + (2-1.6)^2/1.6 + (0-0.4)^2/0.4 = 0.834$$

表 5-8a) 区间 $[0, 7.5]$ 和 $[7.5, 10]$ 的相依表

	$k=1$	$k=2$	Σ
区间 $[0, 7.5]$	$A_{11} = 2$	$A_{12} = 1$	$R_1 = 3$
区间 $[7.5, 10]$	$A_{21} = 2$	$A_{22} = 0$	$R_2 = 2$
Σ	$C_1 = 4$	$C_2 = 1$	$N = 5$

选定的区间应当合并成一个区间, 因为在自由度 $d=1$, $\chi^2_{\text{计算的}} = 0.836 < 2.706$ ($\alpha=0.1$, $d=1$ 的阈值)。在我们的例子中, 关于 χ^2 的给定阈值, 算法将最终确定具有三个区间的离散化: $[0, 10]$, $[10, 42]$ 和 $[42, 60]$, 其中假定 60 是特征 F 的最大值。我们可以对这些区间赋予码值 1、2 和 3 或描述性语言值“低”、“中”和“高”。

不可能出现附加的合并, 因为 χ^2 检验将显示区间之间的显著不同。例如, 如果我们试图合并区间 $[0, 10]$ 和 $[10, 42]$, 相依表将如表 5-8b 所示。期望频率将变成 $E_{11} = 25/9$, $E_{12} = 20/9$, $E_{21} = 20/9$, $E_{22} = 16/9$, $\chi^2_{\text{计算的}} = 5.271 > \chi^2_{\text{查表的}} = 2.706$ 。结论是, 两个区间之间存在显著差别, 不建议合并。

表 5-8b) 区间 $[0, 10]$ 和 $[10, 42]$ 的相依表

	$k=1$	$k=2$	Σ
区间 $[0, 10.0]$	$A_{11} = 4$	$A_{12} = 1$	$R_1 = 5$
区间 $[10.0, 42.0]$	$A_{21} = 1$	$A_{22} = 3$	$R_2 = 4$
Σ	$C_1 = 5$	$C_2 = 4$	$N = 9$

5.4 特征提取、选择和构造

特征提取、选择和构造是预处理任务, 并且独立于数据挖掘。这样做有多个理由。第一, 它可以进行一次, 并且可以在其后所有的数据挖掘任务中使用。第二, 它使用的评估度

量的计算开销通常比数据挖掘算法小,因此它能够处理的数据量比数据挖掘大。第三,它是离线工作的。因此,如果必要的话,可以尝试许多不同的算法。特征提取、选择和构造可以看作三个独立的问题,每一个问题都相当有难度。然而,除了是主要的预处理任务之外,它们还有一些其他共性:①如前所述,它们都试图实现相同的数据归约目标,②它们都需要某种标准,确保结果数据使得挖掘算法效率更高,③它们的有效性需要从多方面度量,如数据的压缩量、压缩数据的相关性,以及(如果可能的话)它们对数据挖掘算法的直接影响。

特征提取、选择和构造可以组合使用。在许多情况下,特征构造用新构造的、更具表达性的特征增加特征的数量,但这样可能包括冗余特征。特征选择可以自动地帮助归约这些多余的特征。可能的组合有:特征选择后随特征提取,特征构造后随特征选择。到底如何使用取决于应用的多个因素,如数据挖掘的期望结果、所用的数据挖掘算法等。

在下面的内容中,我们将分别介绍特征提取、选择和构造。对于每个问题,第一,我们介绍基本概念;第二,讨论一些代表性算法;第三,使用一个简单的数据集展示每种方法的输入和输出,解释算法并进行总结。接下来,我们给出一些使用特征提取、选择和构造的应用。最后,我们用一些可能的进一步研究结束本章的介绍。

5.4.1 特征提取

特征提取是一个过程,它通过某种函数映射从原有的特征提取一个新的特征集(Wyse, Dubes 和 Jain, 1980)。假设有 n 个特征(或属性) A_1, A_2, \dots, A_n , 特征提取后我们有新特征集 $B_1, B_2, \dots, B_m (m < n)$, $B_i = F_i(A_1, A_2, \dots, A_n)$, 而 F_i 是映射函数。例如,一个映射可能是 $B_1 = c_1 A_1 + c_2 A_2$, 其中 c_1 和 c_2 是系数。为了找到好的变换,通常需要计算密集搜索。特征提取的目标是通过某种变换,根据某种性能度量找到一个最小的新特征集。因此,主要的研究问题概述如下。

性能度量研究什么是评估提取的特征的最合适的度量。性能评估的关键是确保变换后提取的特征保持原数据的某些特点。因此,应当选择什么度量在某种程度上取决于需要进行特征提取的应用。通常的数据挖掘任务,如聚类或分类可能对确定性能度量具有很不相同的约束。对于分类,数据具有类标号,训练集上的预测准确率可以作为一种性能度量。对于聚类,数据没有类标号,必须借助于其他度量,如簇内/簇间相似性、数据的方差等。

变换研究将原特征映射到新特征的方法。变换的主要目的是找到一种方法,以更简洁的形式表示原来的数据。在数据挖掘背景下,变换可以定义为找出比原特征更少的新特征,或者找出某些更容易实现可视化和操纵的新特征。不同的映射可以用于特征提取。一般地,映射可以分为线性变换和非线性变换。某些变换只能用于某种类型的数据。通常遇到的情况是数据是否被标记。这样,可以从两个角度对变换分类:线性有标号的、线性无标号的、非线性有标号的和非线性无标号的。许多数据挖掘技术都可以用于变换。例如,EM(期望最大化)、k-均值和k-中心点可以用于无标号数据,多层感知器可以用于有标号的非线性数据。

新特征数考察确定最小新特征数的方法。这看起来是一个容易解决的问题。就像数据可视化,最直观的解决方法是绘制数据的三维模型。然而,当数据的原始维度很高时(超过20维),不大可能具有数据的三维模型的全面视图。我们的目标是创建最小的新特征集,这里的实际问题是有多新特征可以确保变换后的数据保持“真正的本质”。解决该问题的一些常见方法是:①根据以往的经验,在某些变量上主观地确定一个阈值,用于性能度量,从而确定新特征的个数。②根据某种客观度量(如预测准确率)自动地确定特征数。

可以利用数据特征作为选择性能度量、新特征数和变换的标准。除类标号之外,数据特

征可以具有多种类型：连续的、标称的、二元的、混合的。特征提取可能具有多种用途：用于进一步处理的维度归约(Liu 和 Motoda, 1998[7])，可视化(Fayyad, Grinstein 和 Wierse, 2001[8])，用于提升某些数据挖掘算法的复合特征(Liu 和 Setiono, 1997[9])。然而，特征提取确实需要某些开销。最明显的开销如下：

1) 搜索满足性能标准的新特征是非常耗时的。这种缺点通常可以忍受，因为特征提取只是定期执行以得到映射。数据挖掘最频繁执行的是对数据使用映射，然后使用新特征发现有价值的模式。

2) 必须保留原来的特征。换句话说，特征提取并不减少数据源的特征数。如果数据收集的开销很大，那么这种特性是不应当的。在 5.5 节，我们将讨论可以降低原数据的维度的特征选择方法。

特征映射算法：函数映射可以用多种方法实现。在这里，我们提供两个样例算法，解释它们如何处理特征提取。一个带有一些数据的简单例子如表 5-9 所示。

- **前馈神经网络。**前馈神经网络可以用来提取新特征(Setiono 和 Liu, 1997)。特别地，具有一个隐藏层的多层感知器被用于特征提取。其基本思想是使用隐藏单元作为新提取的特征。让我们考察它如何处理特征提取的三个主要问题。第一个问题是如何评估新特征的性能。估计预测准确率，并用它作为性能度量。这要求应该标记数据所属的类。我们选择会使预测准确率达到最佳的提取特征。第二个问题是将原特征映射到新特征。在这种情况下，它是从输入单元到隐藏单元的非线性映射。第三个问题是如何确定新特征的个数。显然，后两个问题与神经网络的拓扑结构密切相关。设计了两个算法用于构造具有最少隐藏单元(即最少特征数)，并且输入层和隐藏层之间具有最少连接的网络：网络构造算法极度节俭地添加一个隐藏单元，以提高预测准确率；如果不影响预测准确率，网络剪枝算法慷慨地剪去输入层和隐藏层之间的冗余连接。
- **主成分分析(PCA)。**这是一种经典技术，其中 n 个原特征被 m 个新特征取代，这些新特征是原特征的线性组合。我们来看一下如何使用 PCA 确定性能度量、变换和新特征数。其基本思想是很简单的：通过这样一些线性组合形成 m 维($1 < m < n-1$)投影，这些线性组合最大化样本方差，与已经选取的所有线性组合不相关。这里的目标是通过从原特征到新提取的特征的线性映射捕获数据中的固有变异性。特殊地，性能度量是样本方差；新特征数 m 由 m 个主成分确定，这些主成分捕获的方差满足预先确定的阈值；变换是线性组合。PCA 并不要求数据具有类标号。 m 个主成分的搜索可以转换为找出数据的协方差矩阵的与 m 个最大本征值相关联的 m 个本征向量(Hand, Mannila, Smyth 和 Uthurusamy, 2001[10])。现在，我们使用频繁引用的鸢尾花数据集来解释 PCA。关于鸢尾花数据集的更多细节在第 6 章给出。

例子：鸢尾花数据是 150×4 的矩阵，其中有 150 行(或实例)和 4 个连续特征。它的协方差矩阵是 4×4 矩阵(如表 5-9 中所示)，其中每个特征值都已经规范化到 $[0, 1]$ 区间。4 个本征值按降序排列显示在表 5-9 中。我们在所有 n 个本征值上计算 m 个最大本征值的比例。前两个的和是 0.95801。也就是说，95% 的方差被前两个主成分捕获。可以使用两个对应的本征向量将原来的 4 维数据变换为两个特征上的新数据：令 M 是 4×2 矩阵，它由两个本征向量组成； D 是原鸢尾花数据；新数据 $D' = DM$ 是二维数据。更多细节以及该方法与其他方法的比较可以在[7]中找到。

表 5-9 鸢尾花数据：协方差矩阵、有序的本征值和它们的比例

协方差矩阵				本征值	比 例
1	-0.1094	0.8718	0.818	2.9108	0.7277
-0.1094	1	-0.4205	-0.3565	0.9212	0.2303
0.8718	-0.4205	1	0.9628	0.1474	0.0368
0.818	-0.3565	0.9628	1	0.0206	0.0052

5.4.2 特征选择

特征选择不同于特征提取，它不产生新特征。它是一个过程，用于从有 N 个特征的原集合中选择 M ($M < N$) 个特征的子集，使得按照某种标准，特征空间的降维是最优的 (Blum 和 Langley, 1997[11])。特征选择在机器学习中的作用是①降低特征空间的维度，②加快学习算法的执行速度，③提高分类算法的预测准确率，④提高学习结果的可理解性。

5.4.3 特征构造

特征构造是一个过程，它通过推断或创建附加的特征来发现特征之间联系的缺失信息和扩展特征空间 (Liu 和 Motoda, 1998)。假设有 n 个原特征 A_1, A_2, \dots, A_n ，进行特征构造后，我们可能有 m 个附加的特征 $A_{n+1}, A_{n+2}, \dots, A_{n+m}$ 。例如，新特征 A_k ($n < k < n+m$) 可以通过在原特征集的 A_i 和 A_j 上执行逻辑操作得到。再看另一个例子。一个二维问题 (例如， A_1 为宽度， A_2 为长度) 可以通过构造面积维 B_1 转换为一个一维问题 (B_1 为面积)。所有新构造的特征都用原来的特征定义。这样，本质上并没有通过特征构造增加新信息。特征构造试图提高原来特征的表达能力。通常，新特征集的维度扩大了，比原特征集的维度高。结果，它不可能直接减少特征数。然而，构造新特征之后，许多特征都成为多余的。直观地，为了构造新特征，可能需要搜索指数多个原特征的组合，并且并非所有的组合都是需要的和有用的。人工构造特征是很困难的。

特征构造的目标是自动地将原来的表示空间变换为可以更好地实现数据挖掘目标的新空间：提高准确率、容易理解、真实的簇、揭示隐藏的模式，等等。特征构造的主要研究问题如下：

- 1) 如何构造新特征？
- 2) 如何为特征构造选择和设计算子？
- 3) 如何使用算子有效地构造新特征？
- 4) 如何度量和选择有用的新特征？

5.5 缺失数据及其处理方法和技巧

在分析数据时，常常会发现每个实例的数据并非总是完整的，会缺失一些数据。在某些情况下，缺失的数据量可能很小，而在另一些情况下，缺失的数据可能很多。本节我们将处理有关缺失数据的如下问题：

- 什么是缺失数据？
- 为什么缺失数据是重要的？
- 缺失数据的主要原因是什么？
- 缺失数据或缺失机制有哪些类型？

- 缺失数据怎么办?
- 通用软件包如何处理缺失数据?

5.5.1 什么是缺失数据

缺失数据是其后续分析需要收集但并未进入数据库的数据。

5.5.2 缺失数据的主要原因

缺失数据的原因有很多。本节将列举一些常见的原因,它们的处理方法会在下一节给出。缺失数据的一种常见形式是受访者拒绝回答某一问题。通常因为问题太敏感。敏感问题的例子包括关于健康、收入和非法活动的问题。另一种常见的原因是回答者根本不知道答案。这可能是记忆问题(例如,记不住上次身体检查的日期),或者是理解问题(不理解问题中某个词或结构)。有时,只是没有可用的期望数据。例如,调查表中关于上次肌肉痉挛距今多长时间的问题会使从未发生肌肉痉挛的人对此问题没有任何反应;询问健康维护组织(HMO)会员,评估他们过去12个月参加活动的情况,对于该时间段没有利用HMO设施的人将不会给出可用的回答。

如果使用计算机辅助面试,则缺失数据可能是因为问卷程序错误造成的。例如,对于前面的问题,不应该首先向过去一年不在HMO的应试者问这些问题。数据处理错误也可能造成缺失数据。数据可能没有输入或未正确输入。缺失数据在研究领域也是常见的,那里的数据是在过去逐渐收集的。例如,在测量促进健康减肥的计划中,可能在计划开始和结束以及其后的6和12个月收集关于知识、态度、饮食习惯和体重的数据。在每个数据收集点,人员会随时间减少,实验对象可能逐渐减少。

5.5.3 缺失数据的机制

处理缺失数据方法的有效性在很大程度上取决于缺失机制[12-28]。例如,如果我们知道一个值为什么缺失,那么就能够利用这一信息猜测它。如果我们没有这种信息,则希望缺失机制是可忽略的(ignoreable),这使得我们可以使用假定它不相关的方法。

统计学家已经确定了三类缺失数据。第一种情况是数据完全随机缺失(Missing Completely At Random, MCAR)。这意味值的缺失与它的值或与其他变量的值不相关。当数据是MCAR时,对于每个记录,变量缺失的概率相同。如果值缺失的概率仅依赖于其他变量的值,则称它是随机缺失(Missing At Random, MAR)。使用概率论的术语,如果我们有包含缺失值的变量 Y 和另一个变量 X ,我们说该数据是MAR,如果 $\Pr(Y \text{ 缺失} | Y, X) = \Pr(Y \text{ 缺失} | X)$ 。如果缺失性依赖于缺失的值,则称该数据是非随机缺失(Not Missing At Random, NMAR),并且这是许多统计数据缺失技术(MDT)的一个问题。例如,当我们使用传感器收集数据,而传感器不能检测超过特定阈值的值时就会出现这种情况。

5.5.4 缺失数据的机制——一个人工例子

为了更好地理解缺失机制,考虑表5-10a所示的数据。它列举了一些病人,以及他们的年龄和化验结果。“病人ID”是一个“官僚化”变量,它不用于数据分析。“年龄”(Age)是独立变量,而“化验结果”(Test Result)是依赖变量。

假定化验费用昂贵。研究者可能决定随机地选择只对某些病人进行化验。结果显示在表 5-10b 中。

表 5-10a) 病人数据

病人 ID	年龄 (Age)	化验结果
Pat1	23	1453
Pat2	23	1354
Pat3	23	2134
Pat4	23	2043
Pat5	75	1324
Pat6	75	1324
Pat7	75	2054
Pat8	75	2056

表 5-10b) 具有完全随机缺失数据 (MCAR) 的病人数据

病人 ID	年龄 (Age)	化验结果
Pat1	23	1453
Pat2	23	null
Pat3	23	2134
Pat4	23	null
Pat5	75	1324
Pat6	75	null
Pat7	75	2054
Pat8	75	null

在分析时, 他们只考虑具有化验结果的记录。注意:

$$\Pr(\text{化验结果缺失} \mid \text{Age} = 23) = \Pr(\text{化验结果缺失} \mid \text{Age} = 75)$$

并且

$$\Pr(\text{化验结果缺失} \mid \text{化验结果}) = \Pr(\text{化验结果缺失})$$

这种机制称为 MCAR (数据完全随机缺失)。

现在, 我们假定主要对老年人做化验。这样, 化验结果缺失事实上取决于“年龄”变量。特殊地,

$$\Pr(\text{化验结果缺失} \mid \text{Age} = 23) = 0.5, \Pr(\text{化验结果缺失} \mid \text{Age} = 75) = 0.0$$

这种机制称为 MAR (数据随机缺失)。表 5-10c 中的数据是 MAR 的例子。

现在, 我们假定用于化验的设备不能测量较高的值。这意味化验结果的缺失依赖于缺失值。更形式地,

$$\Pr(\text{化验结果缺失} \mid \text{化验结果} \leq 2000) = 0.0$$

并且

$$\Pr(\text{化验结果缺失} \mid \text{化验结果} > 2000) = 1.0$$

这样, 化验结果看上去如表 5-10d 所示。

表 5-10c) 具有随机缺失数据 (MAR) 的病人数据

病人 ID	年龄	化验结果
Pat1	23	1453
Pat2	23	null
Pat3	23	2134
Pat4	23	null
Pat5	75	1324
Pat6	75	1324
Pat7	75	2054
Pat8	75	2056

表 5-10d) 具有缺失数据的病人数据

病人 ID	年龄	化验结果
Pat1	23	1453
Pat2	23	1354
Pat3	23	null
Pat4	23	null
Pat5	75	1324
Pat6	75	1324
Pat7	75	null
Pat8	75	null

注意, 没有外部的信息, 该数据统计上不同于表 5-10b 中的数据, 因为

$$\Pr(\text{化验结果缺失} \mid \text{年龄}) \neq \Pr(\text{化验结果缺失})$$

如果我们不知道基本依赖 (通常如此), 可能很难分析该数据。

5.6 在决策树归纳中处理缺失数据的例子

算法 C4.5 假定所有属性的所有值都是确定的。但是在一个数据集中,某些样本的某些属性值常常缺失——这种不完整性在实际应用中很常见。出现这种情况可能是因为对于特定的样本,该值无关紧要;或者在收集数据时,该值未记录;或者是由于输入数据的人的错误所导致。处理缺失的值有两种选择:

- 1) 丢弃数据库中具有缺失值的样本。
- 2) 定义一个新算法或修改一个已有算法来处理缺失值。

第一种解决方案虽然简单,但是当样本集中存在大量缺失值时是不可接受的。对于第二种方案,必须解决如下一些问题:

- 1) 如何比较具有不同数目缺失值的两个样本?
- 2) 具有未知值的训练样本不能与具体的测试值相关联,因此它们不能指派到样例的任何子集。在划分时如何处理这些样本?
- 3) 在分类的检验阶段,如果检验在具有缺失值的属性上进行,如何处理缺失值?

在试图为缺失数据寻找解决方案时,这些问题和其他一些问题都会出现。一些可以处理缺失值的分类算法通常用最可能的值填补缺失值,或考虑给定属性上所有值的概率分布。这些方法没有一种总是最好的。

在 C4.5 中,一个可接受的原则是具有未知值的样本根据已知值的相对频率进行分布。设 $\text{Info}(T)$ 和 $\text{Info}_x(T)$ 按前面的方法计算,不同的是只考虑属性值已知的样本。然后,用一个因子 F 合理地调整增益参数。该因子代表给定属性值已知的概率 ($F = \text{数据库中给定属性具有已知值的样本数} / \text{数据集中的样本总数}$)。新的增益标准具有如下形式:

$$\text{Gain}(x) = F[\text{Info}(T)] - \text{Info}_x(T)$$

类似地,可以修改 $\text{Split-info}(x)$, 在划分时,把具有未知值的样本看作一个附加的组群。如果测试 x 具有 n 个输出,则就像该测试把数据集划分成 $n+1$ 个子集一样来计算它的 $\text{Split-info}(x)$ 。这一修改对修改后的 $\text{Gain-ratio}(x)$ 的最终值具有直接影响。

我们用一个例子解释对 C4.5 决策树方法的修改。数据集与前面的数据集类似,唯一的不同是属性 1 的缺失值用“?”表示,如表 5-11 所示。

表 5-11 包含一个缺失值的简单数据库

属性 1	属性 2	属性 3	类
A	70	True	Class1
A	90	True	Class2
A	85	False	Class2
A	95	False	Class2
A	70	False	Class1
?	90	True	Class1
B	78	False	Class1
B	65	True	Class1
B	75	False	Class1
C	80	True	Class2
C	70	True	Class2
C	80	False	Class1
C	80	False	Class1
C	96	False	Class1

属性1的增益参数的计算与前面类似,只是缺失值修正了前面的某些步骤。属性1中的13个案例中的8个属于类1,5个属于类2,因此划分前的熵是:

$$\text{Info}(T) = -8/13 \log_2(8/13) - 5/13 \log_2(5/13) = 0.961 \text{ 位}$$

使用属性1将 T 划分成3个子集(测试 x_1 表示选择 x_1 的3个值A、B或C之一),结果信息由下式给出:

$$\begin{aligned} \text{Info}_{x_1}(T) &= 5/13(-2/5 \log_2(2/5) - 3/5 \log_2(3/5)) + 3/13(-3/3 \log_2(3/3) - 0/3 \log_2(0/3)) \\ &\quad + 5/13(-3/5 \log_2(3/5) - 2/5 \log_2(2/5)) \\ &= 0.747 \text{ 位} \end{aligned}$$

该测试的信息增益用因子 F (在我们的例子中 $F = 13/14$)调整:

$$\text{Gain}(x_1) = 13/14(0.961 - 0.747) = 0.199 \text{ 位}$$

该测试的增益稍微低于先前的值0.216位。然而,划分信息仍然由整个训练集确定并且比较大,因为存在一个附加的关于未知值的类别。

$$\text{Split-info}(x_1) = -(5/13) + 3/13 \log(3/13) + 5/13 \log(5/13) + 1/13 \log(1/13) = 1.876$$

此外,划分的概念也必须加以推广。每个样本关联一个新参数——概率。当一个具有未知值的样例从 T 指派子集到 T_i 时,它属于 T_i 的概率为1,而属于其他子集的概率为0。当一个值未知时,则只能做较弱的概率陈述。因此,C4.5将每个子集 T_i 中的每个(具有缺失值的)样本关联到一个权重 w (表示案例属于这个子集的概率)。为了使解更一般,必须考虑划分前样本的概率(在决策树构造的其后迭代中)并非总是等于1的情况。这样,划分后缺失值的新参数 w_{new} 等于划分前旧参数 w_{old} 乘以样本属于每个子集的概率 $P(T_i)$ 。形式地,

$$w_{\text{new}} = w_{\text{old}} \cdot P(T_i)$$

使用属性1上的测试 x_1 将集合 T 划分成子集后,具有缺失值的记录将出现在所有3个子集中。结果在表5-12中给出。新权重 w_i 将等于5/13、3/13和5/13,因为 w 的初值(旧值)等于1。新的子集在表5-12中。现在,在C4.5中, $|T_i|$ 不再解释为集合 T_i 中的元素个数,而是解释为给定集合 T_i 中所有权重 w 之和。根据表5-12,这些新值为 $|T_1| = 5 + 5/13$, $|T_2| = 3 + 3/13$, $|T_3| = 5 + 5/13$ 。

表5-12 在 x_1 上的测试结果子集(初始集 T 中包含缺失值)

属性2	属性3	类	w
70	True	Class1	1
90	True	Class2	1
85	False	Class2	1
95	False	Class2	1
70	False	Class1	1
90	True	Class1	13-May
T ₁ : (属性1 = A)			
90	True	Class1	13-Mar
78	False	Class1	1
65	True	Class1	1
75	False	Class1	1
T ₂ : (属性1 = B)			
80	True	Class2	1
70	True	Class2	1
80	False	Class1	1
80	False	Class1	1

(续)

属性 2	属性 3	类	w
96	False	Class1	1
90	True	Class1	13-May

$T_3: (\text{属性 } 1 = C)$

如果这些子集被属性 2 和属性 3 上的测试进一步划分, 则这个具有缺失值的数据集的最终决策树如图 5-2 所示。

IfAttribute = A			
Then			
If Attribute2 <= 70			
Then			
Classification = CLASS 1		(2.0/0);	
Else			
Classification = CLASS 2		(3.4/0.4);	
Elseif Attribute1 = B			
Then			
Classification = CLASS 1		(3.2/0);	
Elseif Attribute1 = C			
Then			
If Attribute3 = True			
Then			
Classification = CLASS 2		(2.4/0);	
Else			
Classification = CLASS 1		(3.0/0).	

图 5-2 具有缺失值的数据集 T 的决策树

图 5-2 中的决策树与其他决策树具有相同的结构, 但是由于最终分类的不明确性, 每个决策附加了两个形如 $(|T_i|/E)$ 的参数。 $|T_i|$ 是到达该树叶的样本之和, E 是属于非指定类的样本数。

例如, $(3.4/0.4)$ 意味 3.4 (或 $3 + 5/13$) 个训练样本到达该树叶, 其中 0.4 (或 $5/13$) 个样本不属于指派到该树叶的类。可以用百分数表示参数 $|T_i|$ 和 E :

给定树叶上的案例中 $3/3.4 * 100\% = 88\%$ 将被分到类 2

给定树叶上的案例中 $0.4/3.4 * 100\% = 12\%$ 将被分到类 1

在决策树用于对先前未在数据集中出现的样本分类时(检验阶段), C4.5 采用类似的方法。如果所有的属性值已知, 则该过程十分简单。从决策树的根节点开始, 属性值上的测试决定决策树的遍历, 并且算法将在一个叶节点上结束, 该叶节点将唯一确定测试案例的类 (或属于类的概率, 如果训练集具有缺失值的话)。如果相关测试属性的值未知, 则不能确定测试的输出。此时, 系统将考察该测试的所有可能输出, 并组合结果分类。由于从树的根节点或子树到这些树叶可能存在多条路径, 因此分类的结果可能是类分布, 而不是单个类。当被检验的案例的整个类分布建立后, 则指定具有最高概率的类为预测的类。

5.7 后处理

后处理的常见步骤如下:

知识过滤: 规则截断和后剪枝。如果训练数据包含噪声, 则学习算法将产生涵盖少量训练对象的决策树的树叶或决策规则。原因是学习算法试图将训练对象的子集划分成更小的真

正一致的子集。为了克服这一问题,必须通过(决策树)后剪枝或(决策规则)截断对决策树或决策规则集进行处理。

解释:我们可以直接用获得的知识进行预测,或作为知识库在专家系统中使用。如果是为终端用户执行知识发现过程,则通常文档形式提供产生的结果。另一种可能的方式是将知识可视化,或转换成终端用户可理解的形式。此外,我们还可以检查新知识是否与先前发现的知识相互抵触。在这一步中,我们还可以汇总规则,将它们和为给定任务提供的领域知识相结合。

评估:学习系统根据训练数据集归纳出概念假定(模型)之后,应当对它们进行评估(检验)。对此,有一些广泛使用的标准:分类的准确性、可理解性、计算复杂性等。

知识集成:传统的决策制定系统依赖于单一模型。新的、复杂的决策支持系统组合或提炼由多个通常使用不同的方法产生的模型而得到的结果。这种过程提高了决策的准确性和成功概率。

参考文献

- [1] J. Dougherty, R. Kohavi, and M. Sahami, Supervised and unsupervised discretization of continuous features. In: *Proc Twelfth International Conference on Machine Learning*, Morgan Kaufmann, Los Altos, CA, 1995.
- [2] J.R. Quinlan, Induction of decision trees, *Machine Learning*, pp. 81–106, 1986.
- [3] R. Kerber, ChiMerge: Discretization of numeric attributes, In: *Proc. Tenth National Conference on Artificial Intelligence*, pp. 123–128, MIT Press, 1992.
- [4] U.M. Fayyad and K.B. Irani, On the handling of continuous-valued attributes in decision tree generation, *Machine Learning*, pp. 87–102, 1992.
- [5] David Maxwell Chickering, Christopher Meek, Robert Rounthwaite, Efficient determination of dynamic split points in a decision tree, pp. 91–98, *ICDM 2001*, San Jose, California, USA.
- [6] R. Kerber, Chimerge: Discretization of numeric attributes, In *Proc AAAI-92*, Ninth National Conference Artificial Intelligence, pp. 123–128, AAAI Press/MIT Press, 1992.
- [7] H. Liu, and Motoda, H. (Eds.), *Feature Extraction, Construction and Selection: A data mining perspective*, Kluwer Academic Publishers, Boston: 1998.
- [8] U. Fayyad, G. Grinstein, and A. Wierse, *Information Visualization in Data Mining and Knowledge Discovery*, Morgan Kaufmann, San Francisco 2001.
- [9] H. Liu and R. Setiono, Feature selection via discretization, *IEEE Transactions on Knowledge and Data Engineering*, pp. 642–645, 1997.
- [10] D. Hand, H. Mannila, and P. Smyth, *Principles of Data Mining*, Cambridge, MA: MIT Press, 2001.
- [11] A. Blum and P. Langley, Selection of relevant features and examples in machine learning, *Artificial Intelligence*, pp. 245–271, 1997.
- [12] Statistical Services of University of Texas (2000), *General FAQ #25: Handling Missing or Incomplete Data [Online]*. Available <http://www.utexas.edu/cc/faqs/stat/general/gen25.html>. [September 1, 2001].
- [13] D. Rubin, *Multiple Imputations in Sample Surveys—A Phenomenological Bayesian Approach to Nonresponse, Imputation and Editing of Faulty or Missing Survey Data*, U.S. Department of Commerce, 1–23, 1978.

- [14] Y. Kim, *The Curse of the Missing Data [Online]*, Available <http://209.68.240.11:8080/2ndMoment/978476655/addPostingForm> September 1, 2001.
- [15] N. Nie, C. Hull, J. Jenkins, K. Steinbrenner, and D. Bent, *SPSS*, McGraw-Hill, New York 1975.
- [16] J. Hair, R. Anderson, R. Tatham and W. Black, *Multivariate Data Analysis*, Prentice-Hall, Upper Saddle River, NJ, 1998.
- [17] A. Dempster and D. Rubin, *Overview*, In: W.G. Madow, I. Olkin, and D. Rubin (Eds.), *Incomplete Data in Sample Surveys*, Vol. II: *Theory and Annotated Bibliography*, New York, Academic Press, pp. 3–10, 1983.
- [18] G. Kalton and D. Kasprzyk, *The Treatment of Missing Survey Data Survey Methodology*, pp. 1–16, 1986.
- [19] L. Sande, Hot-deck imputation procedures, In: W.G. Madow and I. Olkin (Eds.), *Incomplete data in sample surveys*, Vol. 3, *Proceedings of the Symposium on Incomplete*, 1983.
- [20] S. Pennell, Cross-sectional imputation and longitudinal editing procedures in the survey of income and program participation, *Technical report*, Institute of Social Research, University of Michigan, Ann Arbor, MI, 1993.
- [21] V. Iannacchione, Weighted sequential hot deck imputation macros, In the *Proceedings of the SAS Users Group International Conference*, San Francisco, CA, pp. 759–763, 1982.
- [22] J. Graham, S. Hofer, and A. Piccinin, Analysis with missing data in drug prevention research. In L.M. Collins and L. Seitz (Eds.), *Advances in Data Analysis for Prevention Intervention Research*, NIDA Research Monograph, Series (#142). Washington, DC and National Institute on Drug Abuse, 1994.
- [23] R. Royall and J. Herson, Robust estimation from finite populations, *Journal of the American Statistical Association*, pp. 883–889, 1973.
- [24] M. Hansen, W. Madow and J. Tepping, An evaluation of model-dependent and probability-sampling inferences in sample surveys, *Journal of the American Statistical Association*, pp. 776–807, 1983.
- [25] D. Rubin, *Multiple Imputations in Sample Surveys, A Phenomenological Bayesian Approach to Nonresponse, Imputation and Editing of Faulty or Missing Survey Data*, U.S. Department of Commerce, pp. 1–23, 1978.
- [26] R. Little and D. Rubin, *Statistical Analysis with Missing Data*, Wiley, New York, 1987.
- [27] A. Dempster, N. Laird, and D. Rubin, Maximum likelihood from incomplete data via the EM algorithm (with discussion), *Journal of the Royal Statistical Society*, B39, pp. 1–38, 1977.
- [28] J. Cohen and P. Cohen, *Applied multiple regression/correlation analysis for the behavioral sciences*, Lawrence Erlbaum Associates, Hillsdale, NJ, 1983.

第6章 数 据 集

6.1 引言

在过去的数年中,数据挖掘研究者从众多领域收集了一些用于研究的数据集。这些年来,其中一些数据集已经成为比较不同机器学习算法的不同特点的标准数据集。不同的数据集揭示不同的问题。在我们考虑新的学习方法时,留意这些问题是有益的。在本章中,我们将介绍其中的一些数据集,这些数据集对于课堂教学是有用的。我们还将讨论这些标准数据集的来源和描述(但某些数据集缺少描述)。我们将讨论的数据集有:

- 1) 隐形眼镜。
- 2) 鸢尾属植物数据库。
- 3) 乳腺癌数据库。
- 4) 工资数据库。
- 5) 信用数据库。
- 6) 住宅数据库。
- 7) 1985 年汽车进口数据库。
- 8) 徽章问题。

6.2 隐形眼镜

名称: 选择合适隐形眼镜的数据库。

来源: J. Cendrowska, PRISM: An algorithm for inducing modular rules, *International Journal of Man-Machine Studies*, 27, 349-370, 1987。

样本数: 24

属性数: 4(都是标称属性)

属性信息:

目标变量——Contact-lenses(隐形眼镜)(3 个类):

- 1) 患者适合用硬隐形眼镜。
- 2) 患者适合用软隐形眼镜。
- 3) 患者不适合用隐形眼镜。

预测变量:

(1) age(患者年龄^①)

a) young(年轻)

b) pre-presbyopic(提前老花)

c) presbyopic(老花)

(2) Spectacle prescription(验光处方)

① 为了方便读者阅读 arff 格式文件,我们介绍变量(属性)时给出它们的中英文。——译者注

- a) myope(近视)
- b) hypermetrope(远视)
- (3) astigmatic(散光)
- a) no(否)
- b) yes(是)
- (4) tear production rate(流泪)
- a) reduced(减少)
- b) normal(正常)

缺失属性值数量: 0

类分布:

- 1) 硬隐形眼镜: 4
- 2) 软隐形眼镜: 5
- 3) 无隐形眼镜: 15

注意: 数据集存储在电子数据表或 arff 格式(Attribute Relation File Format) 文件中。arff 格式的描述在第 8 章中详细给出。以 % 开始的行是注释。文件开始处的注释之后是关系名和定义属性的块。标称属性后是它们可能的取值, 括在花括号中。数值属性后是类型。这里, 我们给出一些 ARFF 格式的数据集。

arff 文件

```
@relation contact-lenses

@attribute age {young, pre-presbyopic,
               presbyopic}
@attribute spectacle-prescrip {myope, hypermetrope}
@attribute astigmatism {no, yes}
@attribute tear-prod-rate {reduced, normal}
@attribute contact-lenses {soft, hard, none}

@data
%
% 24 instances
%
young,myope,no,reduced,none
young,myope,no,normal,soft
young,myope,yes,reduced,none
young,myope,yes,normal,hard
young,hypermetrope,no,reduced,none
young,hypermetrope,no,normal,soft
young,hypermetrope,yes,reduced,none
young,hypermetrope,yes,normal,hard
pre-presbyopic,myope,no,reduced,none
pre-presbyopic,myope,no,normal,soft
pre-presbyopic,myope,yes,reduced,none
pre-presbyopic,myope,yes,normal,hard
pre-presbyopic,hypermetrope,no,reduced,none
pre-presbyopic,hypermetrope,no,normal,soft
pre-presbyopic,hypermetrope,yes,reduced,none
pre-presbyopic,hypermetrope,yes,normal,none
```

presbyopic, myope, no, reduced, none
 presbyopic, myope, no, normal, none
 presbyopic, myope, yes, reduced, none
 presbyopic, myope, yes, normal, hard
 presbyopic, hypermetrope, no, reduced, none
 presbyopic, hypermetrope, no, normal, soft
 presbyopic, hypermetrope, yes, reduced, none
 presbyopic, hypermetrope, yes, normal, none

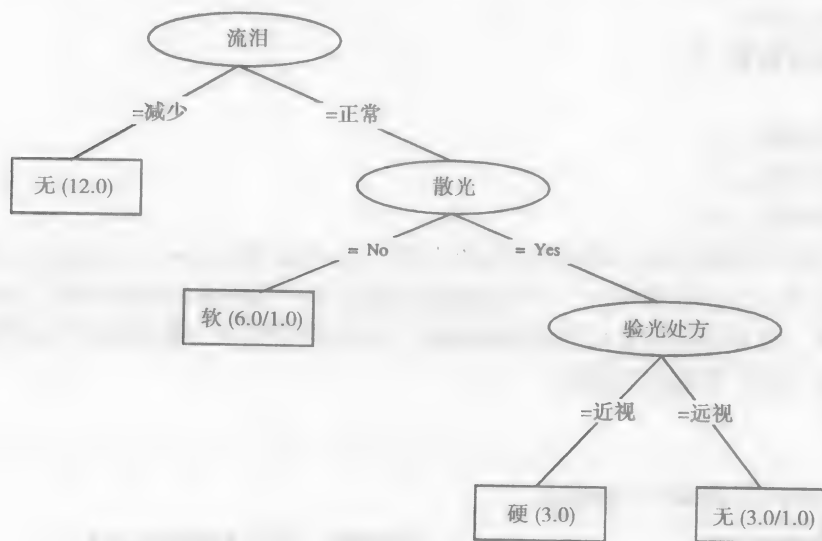


图 6-1 “隐形眼镜”数据集的决策树

6.3 鸢尾属植物数据库

名称: 鸢尾属植物数据库。

来源: 创建者: R. A. Fisher, The use of multiple measurements in taxonomic problems *Annual Eugenics*, 7, Part II, 197-188, 1936, 又见 *Contributions to Mathematical Statistics*, John Wiley, NY, 1950。

相关信息: 这可能是模式识别文献中最著名的数据集。Fisher 的文章是该领域的经典文献, 至今仍被频繁引用(例如, 见 Duda 和 Hart)。该数据集包含 3 个类, 每个类 50 个实例, 其中每个类涉及一种类型的鸢尾属植物。一个类与其他两个类是线性可分的, 而后两个类之间不是线性可分的。

被预测属性: 鸢尾属植物的类。

样本数: 150(3 个类, 每个类 50 个样本)。

属性数: 4 个数值预测属性和类属性。

属性信息:

- 1) Sepal length(cm) (萼片长度)
- 2) Sepal width(cm) (萼片宽度)
- 3) Petal length(cm) (花瓣长度)
- 4) Petal width(cm) (花瓣宽度)
- 5) 类:

- 山鸢尾
- 变色鸢尾
- 维吉尼亚鸢尾

缺失属性值：无

汇总统计：

	最小值	最大值	均值	标准差	类相关
萼片长度	4.3	7.9	5.84	0.83	0.7826
萼片宽度	2	4.4	3.05	0.43	-0.4194
花瓣长度	1	6.9	3.76	1.76	0.949(高!)
花瓣宽度	0.1	2.5	1.2	0.76	0.9565(高!)

类分布：每个类 33.3%。

@RELATION iris

```
@ATTRIBUTE sepallength REAL
@ATTRIBUTE sepalwidth  REAL
@ATTRIBUTE petallength REAL
@ATTRIBUTE petalwidth  REAL
@ATTRIBUTE class        {Iris-setosa,Iris-versicolor,
                          Iris-virginica}
```

@DATA

```
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
4.4,2.9,1.4,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
7.0,3.2,4.7,1.4,Iris-versicolor
6.4,3.2,4.5,1.5,Iris-versicolor
6.9,3.1,4.9,1.5,Iris-versicolor
5.5,2.3,4.0,1.3,Iris-versicolor
6.5,2.8,4.6,1.5,Iris-versicolor
5.7,2.8,4.5,1.3,Iris-versicolor
6.3,3.3,4.7,1.6,Iris-versicolor
4.9,2.4,3.3,1.0,Iris-versicolor
6.6,2.9,4.6,1.3,Iris-versicolor
5.2,2.7,3.9,1.4,Iris-versicolor
6.3,3.3,6.0,2.5,Iris-virginica
5.8,2.7,5.1,1.9,Iris-virginica
7.1,3.0,5.9,2.1,Iris-virginica
6.3,2.9,5.6,1.8,Iris-virginica
6.5,3.0,5.8,2.2,Iris-virginica
7.6,3.0,6.6,2.1,Iris-virginica
4.9,2.5,4.5,1.7,Iris-virginica
7.3,2.9,6.3,1.8,Iris-virginica
```

6.7, 2.5, 5.8, 1.8, Iris-virginica

7.2, 3.6, 6.1, 2.5, Iris-virginica

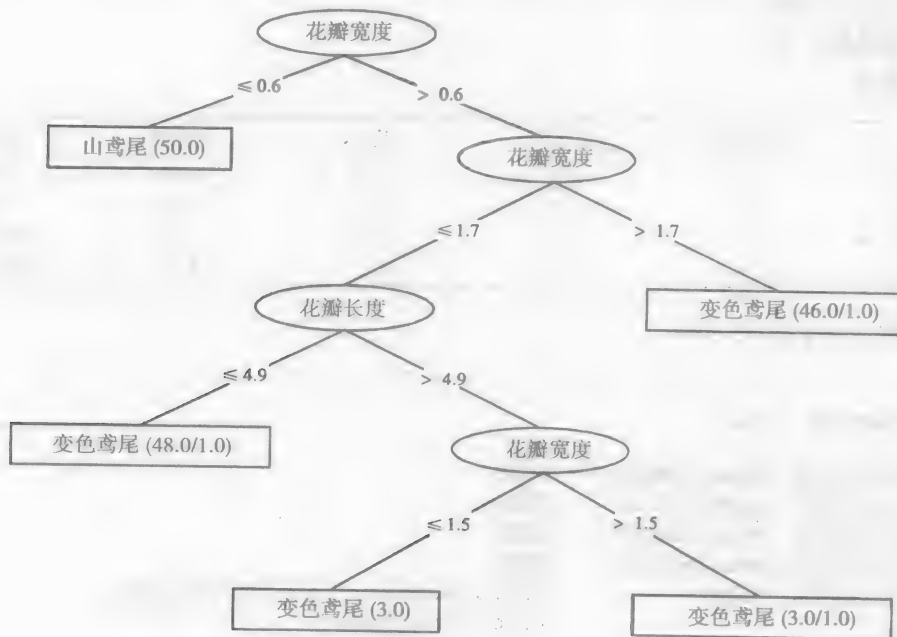
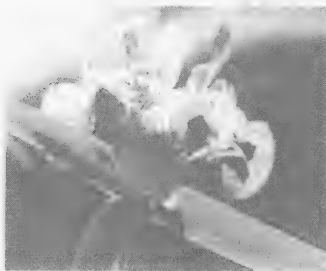


图 6-2a) “鸢尾花”数据集的决策树



山鸢尾



变色鸢尾



图 6-2b) 鸢尾植物的不同种类

6.4 乳腺癌数据库

名称：威斯康星乳腺癌数据库

来源：

- William H. Wolberg 博士(内科医生)
威斯康星大学医院
麦迪逊, 威斯康星
美国
- 捐赠人: Olvi Mangasarian(mangasarian@cs.wisc.edu)
David W. Aha 接收(aha@cs.jhu.edu)

- 日期: 1992 年 7 月 15 日
- 样本数: 699 (1992 年 7 月 15 日数据)
- 属性数: 10 加上类属性
- 属性信息^①: (类属性在最后一列)

序号	属性	域	序号	属性	域
1	样本编码号	ID 号	7	裸细胞核	1-10
2	肿块厚度	1-10	8	温性染色质	1-10
3	细胞大小的均匀性	1-10	9	正常核仁	1-10
4	细胞形状的一致性	1-10	10	有丝分裂	1-10
5	边缘粘连	1-10	11	类	(2 为良性, 4 为恶性)
6	单个上皮细胞大小	1-10			

缺失属性值: 16

在第 1~6 组中有 16 个实例缺失单个属性值, 现在用“?”标记。

类分布:

良性: 458 (65.5%)

恶性: 241 (34.5%)

bewdisc. arff 文件是前面介绍的乳腺癌数据库的离散化版本。

```
@relation breast_cancer
```

```
@attribute Clump_Thickness {0, 1, 2}
@attribute Uniformity_of_Cell_Size {0, 1, 2}
@attribute Uniformity_of_Cell_Shape {0, 1, 2}
@attribute Marginal_Adhesion {0, 1, 2}
@attribute Single_Epithelial_Cell_Size {0, 1, 2}
@attribute Bare_Nuclei {0, 1, 2}
@attribute Bland_Chromatin {0, 1, 2}
@attribute Normal_Nucleoli {0, 1, 2}
@attribute Mitoses {0, 1, 2}
@attribute class {2, 4}
```

```
@data
```

```
1 0 0 0 0 0 1 0 0 2
1 2 2 2 2 2 1 1 0 2
0 0 0 0 0 1 1 0 0 2
2 2 2 0 1 2 1 2 0 2
1 0 0 1 0 0 1 0 0 2
0 0 0 0 0 2 1 0 0 2
0 0 1 0 0 0 1 0 0 2
0 0 0 0 0 0 0 0 2 2
1 1 0 0 0 0 0 0 0 2
0 0 0 0 0 0 1 0 0 2

2 2 2 1 2 2 1 2 2 4
2 2 2 0 0 1 0 0 2 4
1 1 1 2 0 2 1 2 0 4
```

① 数据中属性名的对应中文名见下表中属性 2~11。——译者注

2	1	2	1	1	2	2	2	2	4
1	2	2	2	2	2	2	2	2	4
2	2	2	2	2	2	2	0	0	4
2	2	2	1	2	2	1	2	0	4
2	2	2	0	1	2	1	2	0	4
2	1	2	0	2	0	2	2	2	4
1	1	1	0	2	2	2	0	0	4

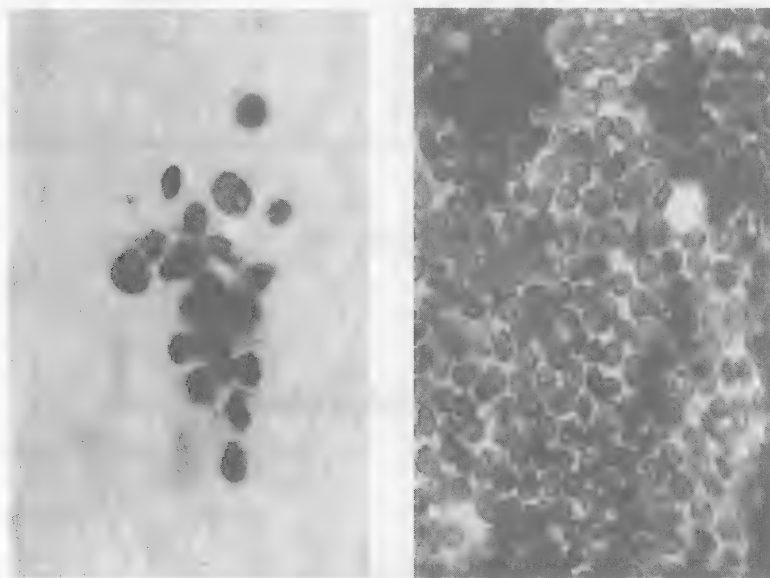


图 6-3a) 从患者胸部抽取的恶性(左)和良性(右)细胞的核仁

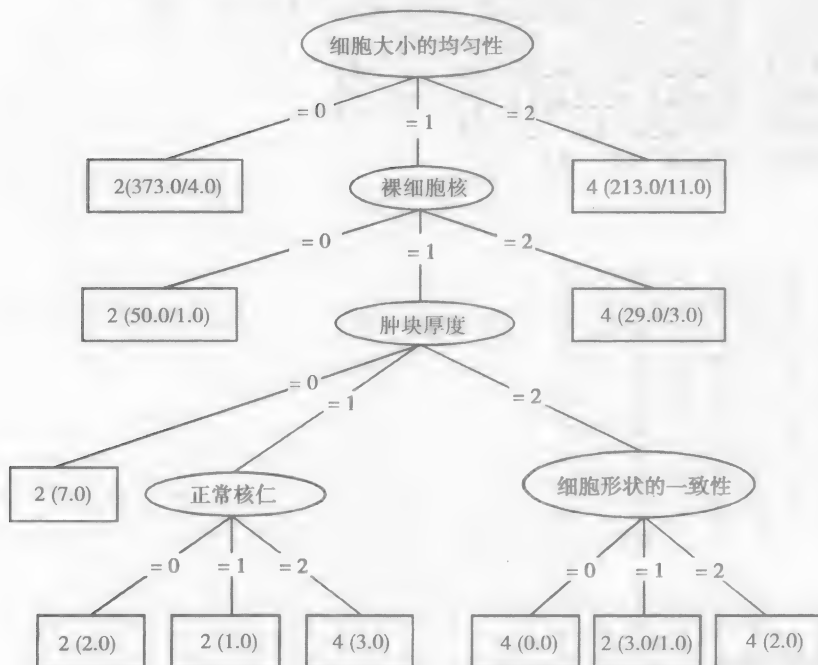


图 6-3b) “乳腺癌”数据集的决策树

6.5 工资数据库

```
@relation wages
```

```
@attribute educ real
@attribute south real
@attribute nonwh real
@attribute hispanic real
@attribute gender real
@attribute married real
@attribute marrfem real
@attribute exper real
@attribute expersq real
@attribute union real
@attribute lnwage real
@attribute age real
@attribute manuf real
@attribute constr real
@attribute manag real
@attribute sales real
@attribute clerical real
@attribute service real
@attribute prof real
@attribute cps85 {Year_1978,Year_1985}
```

```
@data
```

```
12,0,0,0,0,0,0,8,64,0,1.22,25,0,0,0,0,0,1,0,Year_1978
12,0,0,0,1,1,1,30,900,1,1.61,47,0,0,0,0,0,1,0,Year_1978
6,0,0,1,0,1,0,38,1444,1,2.14,49,0,1,0,0,0,0,0,Year_1978
12,0,0,0,0,1,0,19,361,1,2.07,36,0,0,0,0,0,1,0,Year_1978
12,0,0,0,0,1,0,11,121,0,1.65,28,0,0,0,0,0,0,0,Year_1978
8,0,0,0,0,1,0,43,1849,0,1.71,56,0,0,0,0,0,0,0,Year_1978
11,0,0,0,0,0,0,2,4,0,1.1,18,1,0,0,0,0,0,0,Year_1978
15,0,0,0,1,0,0,9,81,0,1.83,29,0,0,1,0,0,0,0,Year_1978
16,0,0,0,1,0,0,17,289,0,0.36,38,0,0,0,0,0,0,1,Year_1978
15,0,0,0,0,1,0,23,529,1,2.15,43,0,0,0,0,0,1,0,Year_1978
15,0,0,0,0,1,0,39,1521,1,1.99,59,1,0,0,0,0,0,1,Year_1978
12,0,0,1,1,1,1,5,25,1,1.7,22,0,0,0,0,0,1,0,Year_1978
11,0,0,1,0,1,0,27,729,1,2.11,43,1,0,0,0,0,0,0,Year_1978
12,0,0,1,0,1,0,29,841,0,1.83,46,1,0,0,0,0,0,0,Year_1978
12,0,0,1,0,0,0,7,49,0,1.31,24,1,0,0,0,0,0,0,Year_1978
12,0,0,0,1,0,0,42,1764,1,2.23,59,0,0,0,0,0,0,1,Year_1978
18,0,0,0,0,1,0,35,1225,1,2.53,58,0,0,0,0,0,0,1,Year_1978
18,0,0,0,1,1,1,31,961,1,2.4,54,0,0,0,0,0,0,1,Year_1978
6,0,0,1,0,1,0,24,576,0,1.25,35,1,0,0,0,0,0,0,Year_1978
14,0,1,0,0,1,0,14,196,1,2.08,33,0,0,0,0,0,0,0,Year_1978
12,0,0,0,1,0,0,40,1600,1,2.01,57,0,0,0,0,0,0,1,Year_1978
12,0,0,0,1,0,0,10,100,0,1.2,27,0,0,1,0,0,0,0,Year_1978
13,0,0,0,1,0,0,3,9,0,1.39,21,0,0,0,1,0,0,0,Year_1978
13,0,0,0,0,0,0,2,4,0,1.23,20,0,0,0,0,1,0,0,Year_1978
15,0,0,1,0,0,0,7,49,0,1.5,27,0,0,0,0,1,0,0,Year_1978
```

10,0,0,0,0,1,0,27,729,0,2.2,43,0,1,0,0,0,0,0,Year_1985
 12,0,0,0,0,1,0,20,400,0,1.7,38,0,0,0,1,0,0,0,Year_1985
 12,0,0,0,1,0,0,4,16,0,1.34,22,0,0,0,1,0,0,0,Year_1985
 12,0,0,0,1,1,1,29,841,0,2.35,47,0,0,0,0,1,0,0,Year_1985
 12,0,0,0,0,1,0,40,1600,1,2.71,58,0,1,0,0,0,0,0,Year_1985

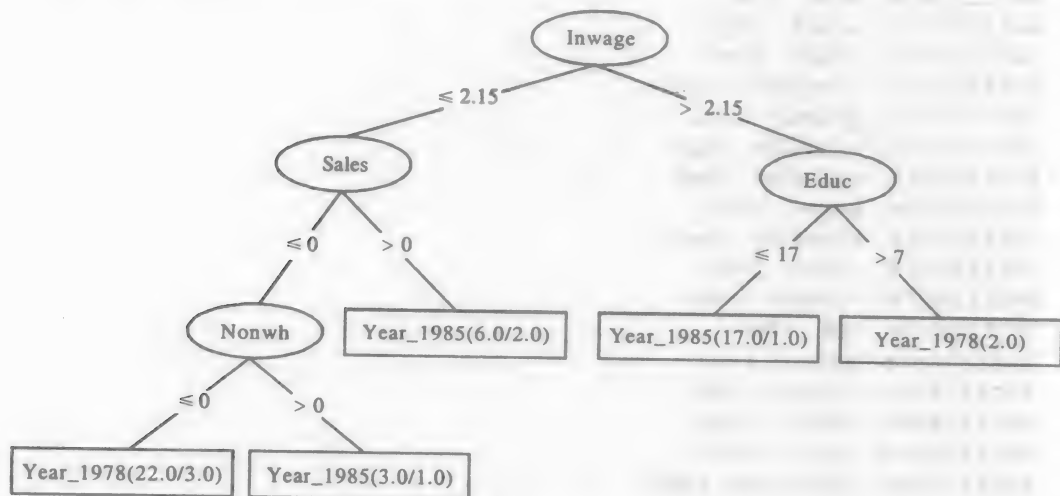


图 6-4 “工资”数据集的决策树

6.6 信用卡数据库

```
@relation credit
```

```
@attribute credit_r {Yes, No}
```

```
@attribute class real
```

```
@attribute pay_week real
```

```
@attribute age real
```

```
@attribute amex real
```

```
@data
```

```
Yes,2,2,2,1
```

```
No,2,1,2,0
```

```
No,4,1,1,1
```

```
Yes,2,2,2,0
```

```
Yes,3,2,1,0
```

```
Yes,1,2,1,1
```

```
Yes,2,2,3,0
```

```
No,2,2,1,0
```

```
No,2,1,1,0
```

```
No,3,1,1,0
```

```
No,5,1,1,0
```

```
No,4,1,1,1
```

```
No,2,2,1,1
```

```
No,2,1,1,1
```

```
No,3,1,1,1
```

No, 2, 1, 2, 1
 No, 4, 1, 1, 1
 No, 2, 2, 1, 0
 No, 3, 1, 1, 1
 No, 5, 1, 1, 0
 No, 4, 1, 1, 1

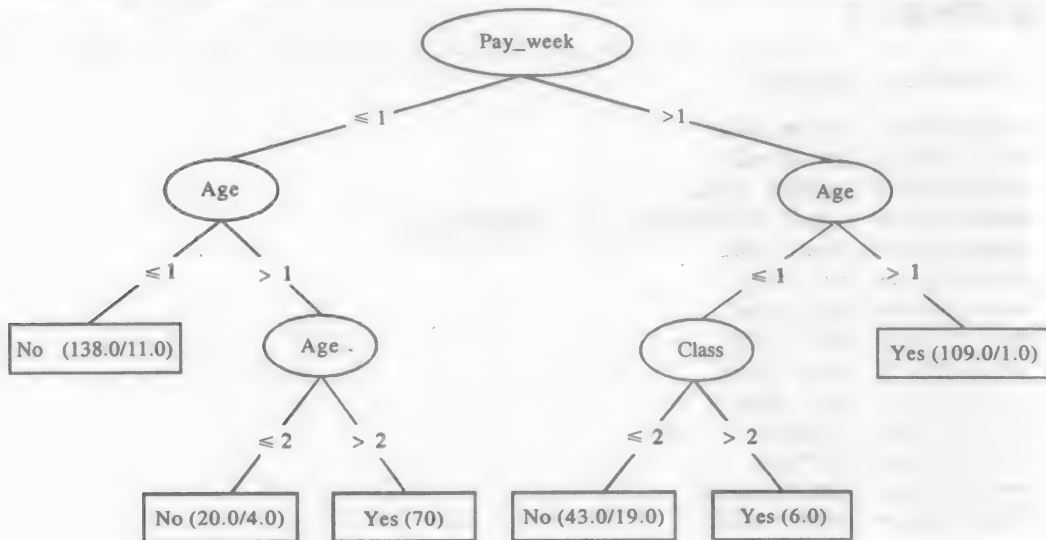


图 6-5 “信用卡”数据集的决策树

6.7 住宅数据库

来源:

1) 起源: 该数据集取自 StatLib 图书馆, 由卡梅基·梅隆大学维护。

2) 创建者: D. Harrison, and D. L. Rubinfeld, Hedonic prices and the demand for clean air, J. Environ. Economics and Management, vol. 5, 81-102, 1978。

3) 日期: 1993 年 7 月 7 日。

相关信息: 关注波士顿郊区住宅价格。

实例数: 200

属性数: 13 个连续属性(包括“类”属性 MEDV), 1 个二元属性。

属性信息:

1) CRIM: 城镇人均犯罪率

2) ZN: 每 25000 平方英尺停车场比例

3) INDUS: 每个城镇非零售业区域的比例

4) CHAS: Charles 河哑变量(如果区域限于河, 则为 1; 否则为 0)

5) NOX: 氮氧化物浓度(每千万)

6) RM: 每个住所的平均房间数

7) AGE: 1940 年之前建立的单元比例

8) DIS: 到波士顿的 5 个工作中心的加权距离

9) RAD: 到干线公路的可达性指数

- 10) TAX: 每一万美元的全值资产税
 11) PTRATIO: 城镇小学生 - 教师比率
 12) B: $1000(b_k - 0.63)^2$, 其中 b_k 是城镇黑人比例
 13) LSTAT: 低地位人口百分比
 14) MEDV: 房价中值(千美元)
 缺失属性值: 无

```
@relation housing
```

```
@attribute crim real
```

```
@attribute zn real
```

```
@attribute indus real
```

```
@attribute chas {Category_1, Category_2}
```

```
@attribute nox real
```

```
@attribute rm real
```

```
@attribute age real
```

```
@attribute dis real
```

```
@attribute rad real
```

```
@attribute tax real
```

```
@attribute ptration real
```

```
@attribute b real
```

```
@attribute lstat real
```

```
@attribute mdev real
```

```
@data
```

```
0.01,18,2.31,Category_1,0.54,6.58,65.2,4.09,1,296,15.3,396.9,4.98,24
0.03,0,7.07,Category_1,0.47,6.42,78.9,4.97,2,242,17.8,396.9,9.14,21.6
0.03,0,7.07,Category_1,0.47,7.19,61.1,4.97,2,242,17.8,392.83,4.03,34.7
0.03,0,2.18,Category_1,0.46,7,45.8,6.06,3,222,18.7,394.63,2.94,33.4
0.07,0,2.18,Category_1,0.46,7.15,54.2,6.06,3,222,18.7,396.9,5.33,36.2
0.03,0,2.18,Category_1,0.46,6.43,58.7,6.06,3,222,18.7,394.12,5.21,28.7
0.09,12.5,7.87,Category_1,0.52,6.01,66.6,5.56,5,311,15.2,395.6,12.43,22.9
0.14,12.5,7.87,Category_1,0.52,6.17,96.1,5.95,5,311,15.2,396.9,19.15,27.1
0.21,12.5,7.87,Category_1,0.52,5.63,100,6.08,5,311,15.2,386.63,29.93,16.5
0.17,12.5,7.87,Category_1,0.52,6,85.9,6.59,5,311,15.2,386.71,17.1,18.9
0.22,12.5,7.87,Category_1,0.52,6.38,94.3,6.35,5,311,15.2,392.52,20.45,15
0.12,12.5,7.87,Category_1,0.52,6.01,82.9,6.23,5,311,15.2,396.9,13.27,18.9
0.09,12.5,7.87,Category_1,0.52,5.89,39,5.45,5,311,15.2,390.5,15.71,21.7
0.63,0,8.14,Category_1,0.54,5.95,61.8,4.71,4,307,21,396.9,8.26,20.4
0.64,0,8.14,Category_1,0.54,6.1,84.5,4.46,4,307,21,380.02,10.26,18.2
0.63,0,8.14,Category_1,0.54,5.83,56.5,4.5,4,307,21,395.62,8.47,19.9
1.05,0,8.14,Category_1,0.54,5.94,29.3,4.5,4,307,21,386.85,6.58,23.1
0.78,0,8.14,Category_1,0.54,5.99,81.7,4.26,4,307,21,386.75,14.67,17.5
0.8,0,8.14,Category_1,0.54,5.46,36.6,3.8,4,307,21,288.99,11.69,20.2
0.73,0,8.14,Category_1,0.54,5.73,69.5,3.8,4,307,21,390.95,11.28,18.2
1.25,0,8.14,Category_1,0.54,5.57,98.1,3.8,4,307,21,376.57,21.02,13.6
0.85,0,8.14,Category_1,0.54,5.97,89.2,4.01,4,307,21,392.53,13.83,19.6
1.23,0,8.14,Category_1,0.54,6.14,91.7,3.98,4,307,21,396.9,18.72,15.2
0.99,0,8.14,Category_1,0.54,5.81,100,4.1,4,307,21,394.54,19.88,14.5
0.75,0,8.14,Category_1,0.54,5.92,94.1,4.4,4,307,21,394.33,16.3,15.6
```

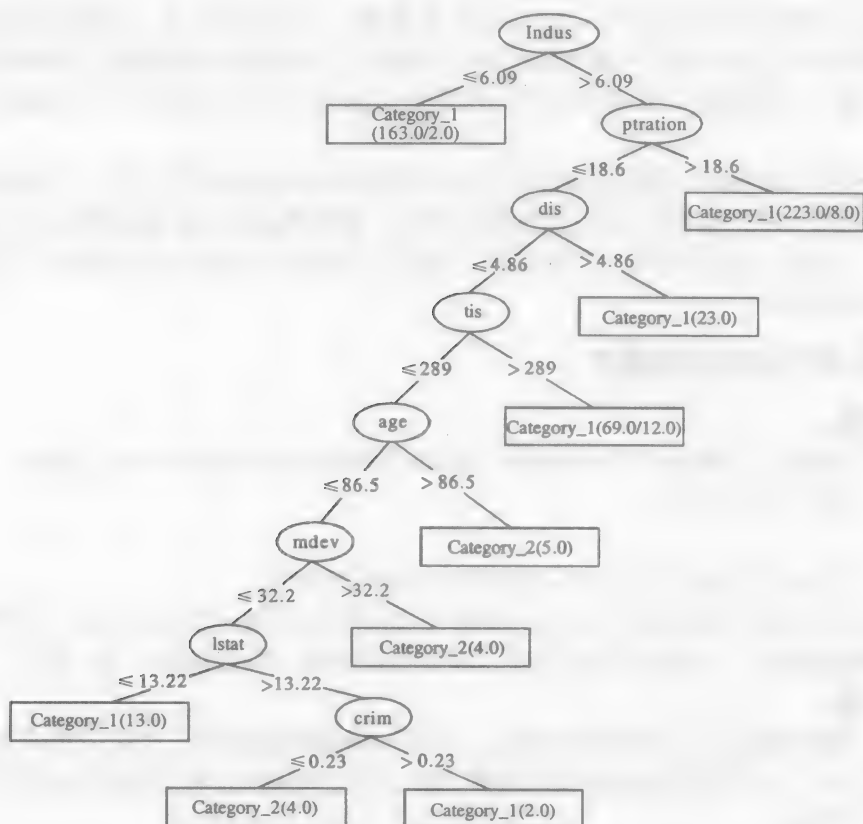


图 6-6 “住宅”数据集的决策树

住宅数据的回归树模型(使用 XLminer)

住宅数据的回归树如图 6-7 所示。该树可以读作：LSTAT(低地位人口百分比)作为第一

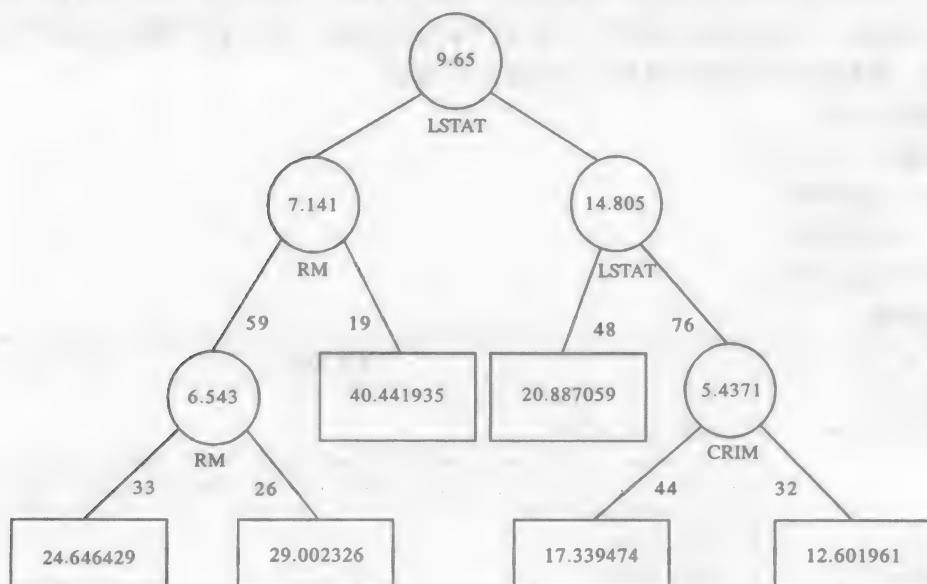


图 6-7 “住宅”数据集的回归树

个划分属性；如果该百分比大于 9.65% (124 个案例)，则 LSTAT 再一次被选为划分属性。现在，如果该百分比小于等于 14.805% (48 个案例)，则 MEDV 被预测为 20.89 美元。因此，我们的第一个规则为：如果 $LSTAT > 9.65\%$ 并且 $LSTAT \leq 14.805\%$ ，则 $MEDV = 20.89$ 美元。

如果 $LSTAT \leq 9.65\%$ ，则我们用 RM (每个住所的平均房间数) 作为下一个划分属性。如果 $RM > 7.141\%$ (19 个案例)，则这些案例的 MEDV 被预测为 40.44 美元 (40.44 美元是另一个叶节点)。因此，我们的第二个规则是：如果 $LSTAT \leq 9.65\%$ 并且 $RM > 7.141\%$ ，则 $MEDV = 40.44$ 美元。

6.8 1985 年汽车进口数据库

来源信息：

创建者/捐赠人：Jeffrey C. Schlimmer (Jeffrey.Schlimmer@a.gp.cs.cmu.edu)

日期：1987 年 5 月 19 日

来源：

- 1) 1985 年进口汽车和卡车说明。1985 年 Ward 汽车年鉴。
- 2) 个人汽车手册，Insurance Services Office, 160 Water Street, New York, NY 10038。
- 3) 保险碰撞报告，公路安全保险公司，Watergate 600, Washington, DC 20037。

相关信息：

描述：该数据集包含 3 种类型的实体：①汽车各种特性的说明。②它的保险风险率。③与其他汽车相比，它的规范化的损失赔偿额。第二个比率对应于该汽车的风险高于相同价格的其他汽车的程度。

开始，汽车被赋予一个与其价格相关联的风险因子符号。之后，如果它的风险更大 (或更小)，则该因子符号向上 (或向下) 调整。保险员称该过程为“用符号表示”。值 +3 指出该汽车是有风险的，而 -3 指出该汽车相当安全。

第三个因子是每个车辆保险年度的相对平均损失赔偿。这个值是对特定分类 (两门小汽车、客货两用车、运动/特殊车辆等) 的所有汽车规范化值，表示每年每辆汽车的平均损失。

注意：该数据库中的多个属性可以用作“类”属性。

实例数：205

属性数：总共 26 个

- 15 个连续属性。
- 1 个整型属性。
- 10 个标称属性。

属性信息：

属 性	属性值域
1. symboling	-3, -2, -1, 0, 1, 2, 3
2. normalized-losses	连续, 从 65 ~ 256
3. make	alfa-romero, audi, bmw, chevrolet, dodge, honda, isuzu, jaguar, mazda, mercedes-benz, mercury, mitsubishi, nissan, peugot, plymouth, porsche, renault, saab, subaru, toyota, volkswagen, volvo
4. fuel-type	柴油, 汽油
5. aspiration	标准, 涡轮
6. num-of-doors	4, 2

(续)

属 性	属性值域
7. body-style	硬顶, 货车, 私家轿车, 后开车门小客车, 敞篷
8. drive-wheels	4 轮驱动, 前轮驱动, 后轮驱动
9. engine-location	前部, 后部
10. wheel-base	连续, 从 86.6 ~ 120.9
11. length	连续, 从 141.1 ~ 208.1
12. width	连续, 从 60.3 ~ 72.3
13. height	连续, 从 47.8 ~ 59.8
14. curb-weight	连续, 从 1488 ~ 4066
15. engine-type	dohc, dohev, 1, ohc, ohcf, ohcv, rotor
16. num-of-cylinders	8, 5, 4, 6, 3, 12, 2
17. engine-size	连续, 从 61 ~ 326
18. fuel-system	1bbl, 2bbl, 4bbl, idi, mfi, mpfi, spdi, spfi
19. bore	连续, 从 2.54 ~ 3.94
20. stroke	连续, 从 2.07 ~ 4.17
21. compression-ratio	连续, 从 7 ~ 23
22. horsepower	连续, 从 48 ~ 288
23. peak-rpm	连续, 从 4150 ~ 6600
24. city-mpg	连续, 从 13 ~ 49
25. highway-mpg	连续, 从 16 ~ 54
26. price	连续, 从 5118 ~ 45400

缺失属性值: (用“?”标记)

属性编号	缺失一个值的实例数
2	41
6	2
19	4
20	4
22	2
23	2
26	4

```

@relation 'autoPrice.names'
@attribute symboling real
@attribute normalized-losses real
@attribute wheel-base real
@attribute length real
@attribute width real
@attribute height real
@attribute curb-weight real
@attribute engine-size real
@attribute bore real
@attribute stroke real
@attribute compression-ratio real
@attribute horsepower real
@attribute peak-rpm real
@attribute city-mpg real
@attribute highway-mpg real

```

```
@attribute class real
```

```
@data
```

```
2,164,99.8,176.6,66.2,54.3,2337,109,3.19,3.4,10,102,5500,24,30,13950
2,164,99.4,176.6,66.4,54.3,2824,136,3.19,3.4,8,115,5500,18,22,17450
1,158,105.8,192.7,71.4,55.7,2844,136,3.19,3.4,8.5,110,5500,19,25,17710
1,158,105.8,192.7,71.4,55.9,3086,131,3.13,3.4,8.3,140,5500,17,20,23875
2,192,101.2,176.8,64.8,54.3,2395,108,3.5,2.8,8.8,101,5800,23,29,16430
0,192,101.2,176.8,64.8,54.3,2395,108,3.5,2.8,8.8,101,5800,23,29,16925
0,188,101.2,176.8,64.8,54.3,2710,164,3.31,3.19,9,121,4250,21,28,20970
0,188,101.2,176.8,64.8,54.3,2765,164,3.31,3.19,9,121,4250,21,28,21105
2,121,88.4,141.1,60.3,53.2,1488,61,2.91,3.03,9.5,48,5100,47,53,5151
1,98,94.5,155.9,63.6,52,1874,90,3.03,3.11,9.6,70,5400,38,43,6295
0,81,94.5,158.8,63.6,52,1909,90,3.03,3.11,9.6,70,5400,38,43,6575
1,118,93.7,157.3,63.8,50.8,1876,90,2.97,3.23,9.41,68,5500,37,41,5572
1,118,93.7,157.3,63.8,50.8,1876,90,2.97,3.23,9.4,68,5500,31,38,6377
1,118,93.7,157.3,63.8,50.8,2128,98,3.03,3.39,7.6,102,5500,24,30,7957
1,148,93.7,157.3,63.8,50.6,1967,90,2.97,3.23,9.4,68,5500,31,38,6229
1,148,93.7,157.3,63.8,50.6,1989,90,2.97,3.23,9.4,68,5500,31,38,6692
1,148,93.7,157.3,63.8,50.6,1989,90,2.97,3.23,9.4,68,5500,31,38,7609
1,110,103.3,174.6,64.6,59.8,2535,122,3.34,3.46,8.5,88,5000,24,30,8921
3,145,95.9,173.2,66.3,50.2,2811,156,3.6,3.9,7,145,5000,19,24,12964
2,137,86.6,144.6,63.9,50.8,1713,92,2.91,3.41,9.6,58,4800,49,54,6479
2,137,86.6,144.6,63.9,50.8,1819,92,2.91,3.41,9.2,76,6000,31,38,6855
1,101,93.7,150,64,52.6,1837,79,2.91,3.07,10.1,60,5500,38,42,5399
1,101,93.7,150,64,52.6,1940,92,2.91,3.41,9.2,76,6000,30,34,6529
1,101,93.7,150,64,52.6,1956,92,2.91,3.41,9.2,76,6000,30,34,7129
```

线性回归模型(使用 Weka 的工具): 当类是连续的时, 可以使用简单线性回归学习一个线性模型, 即一条最接近所有实例的直线的方程。例如, 根据“汽车价格”(autoPrice)数据集, 线性回归工具学习得到:

```
price = -59400 + 79.8symboling + 7.14normalized-losses + 198wheel-base
- 92.5length + 767width + 38.9height + 5.09curb-weight + 49.9engine-size
- 1810bore - 1840stroke + 104compression-ratio + 26.1horsepower
+ 0.753peak-rpm + 18.9city-mpg - 13.5highway-mpg
```

回归树模型(使用 Weka 的工具): 回归树是一棵决策树, 其树叶是落入每个分支的数值类的平均值。这种树的内部节点是非类属性。对于“汽车价格”数据集, 回归树工具学习得到:

```
curb-weight <= 2660 :
| curb-weight <= 2290 :
| | curb-weight <= 2090 :
| | | length <= 161 : price=6220
| | | length > 161 : price=7150
| | curb-weight > 2090 : price=8010
| curb-weight > 2290 :
| | length <= 176 : price=9680
| | length > 176 :
| | | normalized-losses <= 157 : price=10200
| | | normalized-losses > 157 : price=15800
```

```

curb-weight > 2660 :
| width <= 68.9 : price=16100
| width > 68.9 : price=25500

```

模型树：模型树是一棵决策树，其内部节点是非类属性，而叶节点是线性模型。这种树的分支选择运行模型。对于“汽车价格”数据集，模型树工具学习得到：

```

curb-weight <= 2660 :
| curb-weight <= 2290 : LM1
| curb-weight > 2290 :
| | length <= 176 : LM2
| | length > 176 : LM3
curb-weight > 2660 :
| width <= 68.9 : LM4
| width > 68.9 : LM5

```

其中：

```

LM1: price = -5280 + 6.68normalized - losses + 4.44curb - weight
      + 22.1horsepower -85.8city-mpg + 98.6highway - mpg
LM2: price = 9680
LM3: price = -1100 + 91normalized - losses
LM4: price = 9940 + 47.5horsepower
LM5: price = -19000 + 13.2curb - weight

```

6.9 徽章问题

机器学习数据集(<http://archive.ics.uci.edu/ml/>)最好的数据集之一包括称作徽章问题的娱乐数据集。Hakan Kjellerstrand 给出了该问题的一个解决方案。

6.9.1 问题描述

关于该问题没有多少信息(开玩笑!)。每个数据对象包括一个人名和一个符号(+ 或 -)。

名称: ML94/COLT94 徽章问题

信息来源:

- 创建者: Haym Hirsh, 根据 Rob Schapire 的想法
- 捐赠人: Haym Hirsh(hirsh@cs.rutgers.edu)
- 日期: 1994 年 9 月

以往用法: 1994 年机器学习会议和 1994 年计算学习理论会议的提前注册的与会者都收到一个用“+”或“-”标记的徽章。标记根据只有徽章产生器(Haym Hirsh)才知道的某个函数产生,并且只依赖于与会者的名字。与会者的目标是识别用于产生+/-标记的未知函数。

相关信息: 使用程序自动发现未知目标函数的问题之一是决定如何对名字编码,使得可以应用该程序。下面提供的数据以+/-标记后随人名的形式出现。学习系统的用户自行决定如何将该数据转换成系统可以接受的形式(例如,如果你所用的学习器需要特征向量数据,你使用什么属性)。

6.9.2 部分数据

- + Naoki Abe
- Myriam Abramson
- + David W. Aha
- + Kamal M. Ali
- Eric Allender
- + Dana Angluin
- Chidanand Apte
- + Minoru Asada
- + Lars Asker
- + Javed Aslam
- + Haralabos Athanassiou
- + Jose L. Balcazar
- + Timothy P. Barber
- + Michael W. Barley
- Cristina Baroglio
- + Peter Bartlett
- Eric Baum
- + Welton Becket
- Shai Ben-David
- + George Berg
- + Neil Berkman
- + Malini Bhandaru
- + Bir Bhanu
- + Reinhard Blasig
- Avrim Blum
- Anselm Blumer
- + Justin Boyan
- + Carla E. Brodley
- + Nader Bshouty
- Wray Buntine
- Andrey Burago
- + Tom Bylander
- + Bill Byrne
- Claire Cardie
- + Richard A. Caruana
- + John Case
- + Jason Catlett
- + Nicolo Cesa-Bianchi
- Philip Chan
- + Mark Changizi
- + Pang-Chieh Chen
- Zhixiang Chen
- + Wan P. Chiang
- Steve A. Chien

+ Jeffery Clouse
+ William Cohen

Hakan Kjellerstrand 的解决方案如下(用作者自己的话):

首先要认识到, 仅靠原始文件中的数据无法完成我们的机器学习任务。因此, 必须包括其他一些属性, 如长度、计数等。但是, 长度或计数是多少? 当然, 我们可以坐下来并花时间解决它(这或许很好), 但是我个人的目的是使用机器学习工具。

注意: 我假定通过把人名看作字符串来找出解决方案。也就是说, 这个解决方案不能太牵强附会, 利用未在数据集中给出数据的信息(如与会者的身高、年龄或他在会议上做报告的时间)。

需要一点时间考虑, 产生几个可能的解。正如你看到的, 利用我所尝试的属性个数, 我没有直接得到解。

属性名和类型	解 释
name{...}	所有人名(原始形式)
length, 数值	人名长度
even_odd, {0, 1}	人名长度是偶数还是奇数?
first_char_vowel, {0, 1}	第一个字符是元音字母?
second_char_vowel, {0, 1}	第二个字符是元音字母?
vowels, 数值	名字中元音字母的个数
consonants, 数值	辅音字母个数
vowel_consonant_ratio, 数值	元音与辅音字母的比(这是牵强附会?)
spaces, 数值	空格数
dots, 数值	名字中的“.”数, 即名字缩写
words, 数值	词个数, 即名字数, 包括大写缩写
class, {+, -}	徽章标记(原数据给定)

使用 Weka 的结果: Weka 产生如下结果:

```
J48 pruned tree
.....
second_char_vowel = 0: - (84.0)
second_char_vowel = 1: + (210.0)
....
=== Error on training data ===
Correctly Classified Instances 294 100%
.....
=== Confusion Matrix ===
 a b <- classified as
 84 0 | a = -
 0 210 | b = +
....
```

因此, 规则是:

“如果人名的第二个字母是元音字母, 则徽章符号是+, 否则为-。”

第7章 关联规则挖掘

7.1 引言

简言之，关联规则挖掘(也称相依分析)是研究“什么与什么相伴”。例如，医学研究者可能对认识哪些症状伴随哪些已有诊断感兴趣。这些方法又称为购物篮分析(market basket analysis)，因为该问题源于研究顾客事务数据库，以确定购买商品之间的相关性。条码技术的进步使得零售商们可以收集和存储大量销售数据。这些数据被称为购物篮数据。关联规则挖掘可以在包含这种数据项的大型集中发现有趣的关联和相关联系。关联规则展示在给定数据集中频繁一起出现的属性值条件。这种规则的一个例子是98%的购买轮胎和汽车配件的顾客也得到汽车服务。找出这样的规则对于交叉销售(cross marketing)和配送服务是有价值的。关联规则也用于其他应用，如通过识别故障前发生的事件预测通信网络故障。

7.2 事务数据库中关联规则的自动发现

顾客事务细节信息的可用性是开发自动发现存放在数据库中商品之间关联的技术的动力。一个例子是超市中使用条码扫描器收集的数据。这种购物篮数据库由大量事务记录组成。每个记录列出了顾客一次购物交易所购买的所有商品。经理想要知道是否某些商品总是一起销售。他们可能使用这些数据来改善商店布局、优化商品陈列，他们也可能使用这些信息用于交叉销售、促销、分类设计和基于购买模式识别顾客组群。关联规则用形如“if-then”的语句形式提供这类信息。这些规则从数据中得到，并且与逻辑if-then规则不同，关联规则本质上是概率规则。图7-1是这种数据集和某些可能的关联的一个例子。

Tid	商品
1	面包、牛奶
2	面包、尿布、啤酒、鸡蛋
3	牛奶、尿布、啤酒、可乐
4	面包、牛奶、尿布、啤酒
5	面包、牛奶、尿布、可乐

{尿布}→{啤酒}
{牛奶, 面包}→{鸡蛋, 可乐}
{啤酒, 面包}→{牛奶}

图7-1 项集和关联规则

可能的关联还有很多，而我们感兴趣的是找出强关联。为此，我们需要一些度量。这些度量将在7.2.1节讨论。

支持度和置信度

除前件(“if”部分)和后件(“then”部分)外，每个关联规则还有两个数，表达规则的不确

定程度。在关联分析中,前件和后件都是不相交的(不含公共项)项的集合(称作项集)。第一个数称为规则的支持度(support)。支持度是包含规则前件和后件中所有项的事务个数。(有时,支持度用这些事务占数据库中全部记录的百分比表示。)

另一个数称为规则的置信度(confidence)。置信度是包含前件和后件中所有项的事务(即支持度)与包含前件中所有项的事务的比。为了更具体地解释,我们考虑图 7-1 中的数据集合。

假设 σ 表示项集出现的支持度计数或频度。

例如, $\sigma(\{\text{牛奶}, \text{面包}, \text{尿布}\}) = 2$

设 s 表示包含一个项集的事务所占的比例。

例子: $s(\{\text{牛奶}, \text{面包}, \text{尿布}\}) = 2/5 = \frac{\sigma(\{\text{牛奶}, \text{面包}, \text{尿布}\})}{|T|}$, 其中 $|T|$ 是事务总数。

关联规则是一个形如 $X \Rightarrow Y$ 的蕴涵表达式,其中 X 和 Y 都是项集。

例子: $\{\text{牛奶}, \text{尿布}\} \Rightarrow \{\text{啤酒}\}$

现在,我们定义规则的支持度和置信度(规则的评估度量)。

$s(X \Rightarrow Y)$ = 包含 X 和 Y 的事务所占的比例

$$s(\{\text{牛奶}, \text{尿布}\} \Rightarrow \{\text{啤酒}\}) = \frac{\sigma(\{\text{牛奶}, \text{尿布}, \text{面包}\})}{|T|}$$

设 c 表示度量“ Y 中的项在包含 X 的事务中出现的频繁性”。

$$\text{例子: } c(\{\text{牛奶}, \text{尿布}\} \Rightarrow \{\text{啤酒}\}) = \frac{\sigma(\{\text{牛奶}, \text{尿布}, \text{面包}\})}{\sigma(\{\text{牛奶}, \text{尿布}\})} = 2/3 = 0.67$$

可能由项集 $\{\text{牛奶}, \text{尿布}, \text{啤酒}\}$ 产生的一些规则如下:

$\{\text{牛奶}, \text{尿布}\} \Rightarrow \{\text{啤酒}\} (s=0.4, c=0.67)$

$\{\text{牛奶}, \text{啤酒}\} \Rightarrow \{\text{尿布}\} (s=0.4, c=1.0)$

$\{\text{尿布}, \text{啤酒}\} \Rightarrow \{\text{牛奶}\} (s=0.4, c=0.67)$

$\{\text{啤酒}\} \Rightarrow \{\text{牛奶}, \text{尿布}\} (s=0.4, c=0.67)$

$\{\text{尿布}\} \Rightarrow \{\text{牛奶}, \text{啤酒}\} (s=0.4, c=0.5)$

$\{\text{牛奶}\} \Rightarrow \{\text{尿布}, \text{啤酒}\} (s=0.4, c=0.5)$

由给定项集产生的所有规则都具有相同的支持度,但是置信度一般不同。

注意: 置信度概念不同于(也无关于)统计推断中的置信区间和置信水平。一种方法是把支持度看作从数据库中随机选择的事务将包含规则前件和后件中所有项的概率,而置信度是给定随机选择的事务包含规则前件中的所有项,该随机选择的事务包含后件中所有项的条件概率。

为了理解算法的复杂性和给定项集可能产生的规则的数目,考虑项集 $\{A, B, C, D, E\}$ 。图 7-2 中存在 $2^5 = 32$ 个节点,并且每个节点元素(项集)都是一个产生规则的可能的候选项集。[1]证明对于 d 个项,我们可以产生 R 个规则,其中 R 由下式计算:

$$R = \sum_{k=1}^d \left[C_d^k \times \sum_{j=1}^{d-k} C_{d-k}^j \right] = 3^d - 2^{d+1} + 1$$

如果 $d=6$,我们可以形成 602 个规则。

现在,我们用一个规模较大的例子来认识该问题的复杂性,并深入讨论下一节的算法。

例 7.1 电子产品销售

All Electronics 零售店的经理想知道哪些商品一起销售。他有一个如表 7-1 所示的数据库。

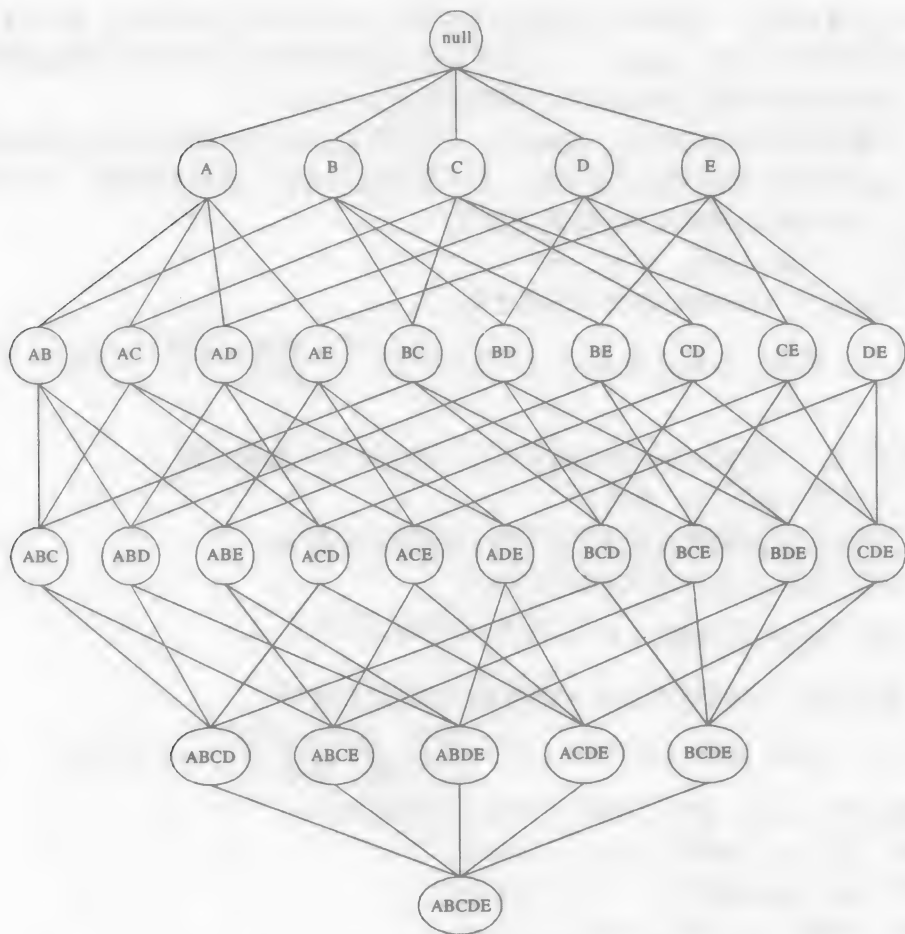


图 7-2 从给定项集产生规则的候选项集

表 7-1 事务数据

事务 ID	商品代码			
1	1	2	5	
2	2	4		
3	2	3		
4	1	2	4	
5	1	3		
6	2	3		
7	1	3		
8	1	2	3	5
9	1	2	3	

这里有 9 个事务。每个事务都记录了一起购买的商品代码。事务 1 是同时购买商品 1、2 和 5。事务 2 是同时购买商品 2 和 4，等等。假设我们想得到该数据库中支持度计数至少为 2（等价于百分比支持度 $2/9 = 22\%$ ）的关联规则。通过枚举，我们可以看出只有下面一些项集的支持度计数至少为 2：

- {1} 支持度计数为 6
- {2} 支持度计数为 7
- {3} 支持度计数为 6
- {4} 支持度计数为 2
- {5} 支持度计数为 2
- {1, 2} 支持度计数为 4

- {1, 3} 支持度计数为 4
- {1, 5} 支持度计数为 2
- {2, 3} 支持度计数为 4
- {2, 4} 支持度计数为 2
- {2, 5} 支持度计数为 2
- {1, 2, 3} 支持度计数为 2
- {1, 2, 5} 支持度计数为 2

注意,一旦我们得到具有所要求的支持度的所有项集的列表,我们就可以通过考察列表中每个项集的所有子集,归纳出满足期望置信度的规则。由于集合的任何子集的出现频率至少与该集一样,因此每个子集也在该列表中。这样,可以直接根据项集的支持度(计数)和该项集的每个子集的支持度(计数)之比计算规则的置信度。仅当置信度超过期望的置信度阈值,我们才保留相应的关联规则。例如,根据项集{1, 2, 5},我们得到如下关联规则:

- {1, 2} \Rightarrow {5} 其置信度 = {1, 2, 5} 的支持计数除以 {1, 2} 的支持计数 = $2/4 = 50\%$
- {1, 5} \Rightarrow {2} 其置信度 = {1, 2, 5} 的支持计数除以 {1, 5} 的支持计数 = $2/2 = 100\%$
- {2, 5} \Rightarrow {1} 其置信度 = {1, 2, 5} 的支持计数除以 {2, 5} 的支持计数 = $2/2 = 100\%$
- {1} \Rightarrow {2, 5} 其置信度 = {1, 2, 5} 的支持计数除以 {1} 的支持计数 = $2/6 = 33\%$
- {2} \Rightarrow {1, 5} 其置信度 = {1, 2, 5} 的支持计数除以 {2} 的支持计数 = $2/7 = 29\%$
- {5} \Rightarrow {1, 2} 其置信度 = {1, 2, 5} 的支持计数除以 {5} 的支持计数 = $2/2 = 100\%$

如果期望的置信度阈值为 70%, 则我们只报告第二、第三和最后一个规则。我们可以看出,产生满足预定支持度和置信度的所有关联规则问题可以分为两步。第一步找出满足支持度要求的所有项集(这些项集称作频繁项集或大项集);然后,根据每个选出的项集产生满足置信度要求的关联规则。对于大部分关联分析数据,计算的困难在于第一步。

7.3 Apriori 算法

尽管已经提出一些产生关联规则的算法,但是最经典的算法是 Agrawal 和 Srikant 的 Apriori(先验)算法(1993)[1]。该算法的基本思想是从只包含一个项的频繁项集(1-项集)开始,递归地产生具有两个项的频繁项集,然后产生具有 3 个项的频繁项集,如此下去,直到产生所有的频繁项集。不失一般性,我们用唯一、相继的(正)整数表示项,并且每个项集中的项以这种项编号的递增序排列。

7.2 节讨论的例子解释了这种记号。在计算中谈及一个项时,我们实际是指这种项编号。

产生频繁 1-项集比较容易。我们需要做的是对每个项计数,看数据库中有多少个事务包含该项。这些事务计数是 1-项集的支持度。丢弃那些支持度低于期望阈值的 1-项集,得

到频繁 1-项集的列表。对于 $k = 2, 3, \dots$, 产生过程按如下方法由频繁 $(k-1)$ -项集得到频繁 k -项集。在频繁 $(k-1)$ -项集列表上进行 $(k-1)$ -项集对的连接运算, 创建 k -项集候选列表。一对 $(k-1)$ -项集被组合在一起, 仅当两个项集的前 $(k-2)$ 项相同。(当 $k=2$ 时, 这意味组合所有可能的 1-项集对。) 如果条件满足, 则该对的连接是一个 k -项集, 它包含前 $(k-2)$ 个公共项和两个非公共项, 每个非公共项来自该对的一个成员。所有的频繁 k -项集一定都在这个候选列表中, 因为频繁 k -项集的每个大小为 $(k-1)$ 的子集都必须是频繁 $k-1$ 项集。然而, 该候选列表中的某些 k -项集可能不是频繁 k -项集。我们需要删除这些候选项集, 创建频繁 k -项集列表。为了识别非频繁的 k -项集, 我们考察每个候选 k -项集的所有大小为 $(k-1)$ 的子集。注意, 我们只需要考察至少包含该候选 k -项集中两个项的 $(k-1)$ -项集。(为什么?) 如果这些大小为 $(k-1)$ 的子集中的某一个不在频繁 $(k-1)$ 项集列表中, 则该候选 k -项集不可能是频繁项集。

我们从候选列表中删除这样的 k -项集。以这种方式处理候选项集列表中的每个项集, 确保扫描结束时候选 k -项集列表被剪裁。然后扫描数据库, 确定频繁 k -项集。增加 k , 递归地重复这一过程, 当候选列表为空时停止。该算法有效性的关键点是候选和频繁项集列表的数据结构。算法的最初版本使用散列树, 但是后来提出了一些改进这种结构的提议。还有一些其他算法, 在实践中运行速度比 Apriori 算法快。从非技术的意义来讲, 什么是“置信度”? 如何保证我们得到的规则是有意义的? 从统计学的角度考虑这一问题, 我们能够回答“我们发现的关联规则实际上只是偶然出现的吗?”

使用 Apriori 算法的例子

考虑表 7-2 所示的事务数据库。

表 7-2 事务数据库

顾 客	商 品
C1	牛奶, 鸡蛋, 面包, 薯片
C2	鸡蛋, 爆米花, 薯片, 啤酒
C3	鸡蛋, 面包, 薯片
C4	牛奶, 鸡蛋, 面包, 爆米花, 薯片, 啤酒
C5	牛奶, 面包, 啤酒
C6	鸡蛋, 面包, 啤酒
C7	牛奶, 面包, 薯片
C8	牛奶, 鸡蛋, 面包, 黄油, 薯片
C9	牛奶, 鸡蛋, 黄油, 薯片

- 首先, 我们识别哪些商品频繁地一起购买(这些称为频繁项集)。
- 然后, 我们从频繁项集推导出强规则。
- 假定最小支持度(min_sup) $s = 30\%$ (或至少在 3 个事务中)。
- 首先, 我们扫描数据库, 识别所有单个项(1-项集)和它们的支持度。它们称作候选 1-项集, 记作 C_1 (见表 7-3)。
- 然后, 我们选择其支持度大于或等于 min_sup 的项。这些被称为频繁 1-项集, 并记作 L_1 (见表 7-4)。

表 7-3 候选 1-项集 C_1

项 集	支持度计数
{牛奶}	6
{鸡蛋}	7
{面包}	7
{爆米花}	2
{黄油}	2
{薯片}	7
{啤酒}	4

表 7-4 频繁 1-项集 L_1 和支持度

项 集	支持度计数
{牛奶}	6
{鸡蛋}	7
{面包}	7
{薯片}	7
{啤酒}	4

我们识别了所有的频繁 1-项集。

- 下一步, 我们需要做类似的工作, 识别所有的频繁 2-项集。
- 首先, 我们产生所有可能频繁的 2-项集。它们称作候选 2-项集或 C_2 (见表 7-5)。
- 这可以通过从 L_1 产生所有可能的 2-项集来实现 (注意, 我们不需要考虑非频繁的 1-项集)。
- 扫描数据库, 确定 C_2 中每个项集的支持度。结果显示在表 7-6 中。

表 7-5 候选 2-项集 C_2

项 集
{牛奶, 鸡蛋}
{牛奶, 面包}
{牛奶, 薯片}
{牛奶, 啤酒}
{鸡蛋, 面包}
{鸡蛋, 薯片}
{鸡蛋, 啤酒}
{面包, 薯片}
{面包, 啤酒}
{薯片, 啤酒}

表 7-6 候选 2-项集 C_2 和它们的支持度

项 集	支持度计数
{牛奶, 鸡蛋}	4
{牛奶, 面包}	5
{牛奶, 薯片}	5
{牛奶, 啤酒}	2
{鸡蛋, 面包}	5
{鸡蛋, 薯片}	6
{鸡蛋, 啤酒}	3
{面包, 薯片}	5
{面包, 啤酒}	3
{薯片, 啤酒}	2

- 我们从 C_2 中选择那些满足 \min_sup 的项集。得到的表 L_2 如表 7-7 所示。
- 重复以上过程, 直到不再有候选 (或频繁) 项集为止。
- 这样, 该过程形如 $C_1 \rightarrow L_1 \rightarrow C_2 \rightarrow L_2 \rightarrow C_3 \rightarrow L_3 \rightarrow \dots$
- 在继续进行之前, 我们先考虑一种称为 Apriori 性质的重要性质。
- **Apriori 性质:** 一个项集是频繁的, 则它的所有非空子集都必须是频繁的。换句话说, 如果一个项集不是频繁的, 则它的所有超集都不可能是频繁的。
- 在下面的处理过程中, 我们使用该性质。
- 从 L_2 计算 C_3 :
 - 连接两个频繁 2-项集 l_1 和 l_2 (或对它取并), 产生一个候选 3-项集 (见表 7-8)。
 - 仅当 l_1 的第一个项与 l_2 的第一个项相同时, 我们才连接它们。这里, 我们假定项集中的项已经按某种顺序排序, 通常按字母序排序。(注意, 在我们使用的例子中, 项并未按字母序排序。但是, 在所有项集中, 项的次序是一致的。)
 - 例如, 我们连接 {牛奶, 鸡蛋} 和 {牛奶, 面包}, 产生 {牛奶, 鸡蛋, 面包}, 但是并不连接 {牛奶, 鸡蛋} 和 {面包, 薯片}。

表 7-7 具有要求的最小支持度的频繁 2-项集 L_2

项 集	支持度计数
{牛奶, 鸡蛋}	4
{牛奶, 面包}	5
{牛奶, 薯片}	5
{鸡蛋, 面包}	5
{鸡蛋, 薯片}	6
{鸡蛋, 啤酒}	3
{面包, 薯片}	5
{面包, 啤酒}	3

表 7-8 候选 3-项集 C_3

项集
{牛奶, 鸡蛋, 面包}
{牛奶, 鸡蛋, 薯片}
{牛奶, 面包, 薯片}
{鸡蛋, 面包, 薯片}
{鸡蛋, 面包, 啤酒}
{鸡蛋, 薯片, 啤酒}
{面包, 薯片, 啤酒}

- 确定支持度。
- 我们需要扫描数据库来确定支持度。
- 如果 $|C_3|$ 很大, 确定支持度需要花费很多时间。
- 因此, 我们需要删除 C_3 中不可能用于产生 L_3 的项集。
- 这一过程称作剪枝过程。
- 如果候选 3-项集 I 至少有一个 2-项集子集不是频繁的, 则我们可以从 C_3 中删除 I 。(为什么?)
- 在上面的 C_3 中, 可以将 {鸡蛋, 薯片, 啤酒} 从 C_3 中删除, 因为 {薯片, 啤酒} 不是频繁的 (不在 L_2 中)。{面包, 薯片, 啤酒} 也不是频繁的。(验证其他候选项集都不包含非频繁的 2-项集子集。)
- 剪枝后, 扫描数据库, 确定支持度。结果显示在表 7-9 中。
- 由 C_3 确定 L_3 。见表 7-10。

表 7-9 剪枝后的候选 3-项集 C_3

项 集	支持度计数
{牛奶, 鸡蛋, 面包}	3
{牛奶, 鸡蛋, 薯片}	4
{牛奶, 面包, 薯片}	4
{鸡蛋, 面包, 薯片}	4
{鸡蛋, 面包, 啤酒}	2

表 7-10 具有要求的支持度的频繁 3-项集 L_3

项 集	支持度计数
{牛奶, 鸡蛋, 面包}	3
{牛奶, 鸡蛋, 薯片}	4
{牛奶, 面包, 薯片}	4
{鸡蛋, 面包, 薯片}	4

- 由 L_3 计算 C_4 (结果在表 7-11 中)。
- 同样, 在连接两个频繁 3-项集时, 我们首先检查它们的前两个项是否相同。仅当它们的前两个项相同时才连接它们。
- 由于 {牛奶, 鸡蛋, 面包, 薯片} 的所有 3-项集子集都在 L_3 中, 因此我们保留它。
- 扫描数据库并确定支持度。结果在表 7-12 中。

表 7-11 候选 4-项集 C_4

项集
{牛奶, 鸡蛋, 面包, 薯片}

表 7-12 候选 4-项集 C_4 和它的支持度

项集	支持度计数
{牛奶, 鸡蛋, 面包, 薯片}	3

- 由 C_4 确定 L_4 (见表 7-13)。

表 7-13 具有要求的支持度的频繁 4-项集 L_4

项集	支持度计数
{牛奶, 鸡蛋, 面包, 薯片}	3

- 由 $C_5 = \phi$, 停止。频繁项集为 $L = L_1 \cup L_2 \cup L_3 \cup L_4$ 。
- 推导强规则
 - 考虑频繁 3-项集 {鸡蛋, 面包, 薯片}。
 - 由于这 3 种商品频繁地一起购买, 我们可能能够从这个 3-项集推导出一些规则。
 - 首先, 我们得到非空真子集: {鸡蛋}, {面包}, {薯片}, {鸡蛋, 面包}, {鸡蛋, 薯片}, {面包, 薯片}。
 - 然后, 对于每个子集, 我们形成如下规则:

$R_1: \{鸡蛋\} \Rightarrow \{面包, 薯片\}$

$R_2: \{面包\} \Rightarrow \{鸡蛋, 薯片\}$

$R_3: \{薯片\} \Rightarrow \{鸡蛋, 面包\}$

$R_4: \{鸡蛋, 面包\} \Rightarrow \{薯片\}$

$R_5: \{鸡蛋, 薯片\} \Rightarrow \{面包\}$

$R_6: \{面包, 薯片\} \Rightarrow \{鸡蛋\}$

注意: 规则的右部由 {鸡蛋, 面包, 薯片} —— 规则左部得到。

- 为了确定强规则, 我们计算置信度:

$R_1: \{鸡蛋\} \Rightarrow \{面包, 薯片\}, 4/7$ 或 57.1%

$R_2: \{面包\} \Rightarrow \{鸡蛋, 薯片\}, 4/7$ 或 57.1%

$R_3: \{薯片\} \Rightarrow \{鸡蛋, 面包\}, 4/7$ 或 57.1%

$R_4: \{鸡蛋, 面包\} \Rightarrow \{薯片\}, 4/5$ 或 80%

$R_5: \{鸡蛋, 薯片\} \Rightarrow \{面包\}, 4/6$ 或 66.7%

$R_6: \{面包, 薯片\} \Rightarrow \{鸡蛋\}, 4/5$ 或 80%

- 如果 \min_conf 为 80%, 则选取 R_4 和 R_6 为强规则。
- 如果 \min_conf 为 60%, 则 R_5 也是强规则。
- 这一过程可以概括如下:
 - 对于每个频繁项集 (频繁 1-项集除外) I , 找出 I 的所有非空真子集。
 - 对于 I 的每个子集 s , 形成一个规则 $s \Rightarrow I - s$ 。
 - 对于每个规则 R , 计算它的置信度 $\text{conf}(R) = \text{sup}(I) / \text{sup}(s)$ 。
 - 如果 $\text{conf}(R) \geq \min_conf$, 则选取 R 为强规则。

7.4 缺点

在实践中, 关联规则可能并不像人们期望的那么有用。一个主要缺点是支持度置信度框架常常产生过多的规则。另一个缺点是其中大部分规则是显而易见的。诸如“星期五晚上尿布和啤酒一起购买”的著名故事并不像期望的那样普遍。关联分析需要技巧, 并且正如一些研究者主张的那样, 用更严格的统计学知识处理规则增值将是有益的。

习题

1. 考虑如下频繁 3-项集的集合:

$\{1, 2, 3\}$, $\{1, 2, 4\}$, $\{1, 2, 5\}$, $\{1, 3, 4\}$, $\{2, 3, 4\}$, $\{2, 3, 5\}$, $\{3, 4, 6\}$
假定数据集中只有6个项。

1) 列出由 Apriori 的候选产生过程得到的所有候选 4-项集。

2) 列出 Apriori 算法的候选剪枝步骤之后剩下的所有候选 4-项集。

2. Apriori 算法使用“产生-计数”策略导出频繁项集。大小为 $k+1$ 的候选项集通过连接一对大小为 k 的频繁项集而构造的(称为候选产生步骤)。例如, 候选 $\{P, Q, R\}$ 通过合并频繁项集 $\{P, Q\}$ 和 $\{P, R\}$ 产生。如果发现一个候选的某个子集不是频繁的, 则该候选被剪枝。例如, 如果 $\{Q, R\}$ 是非频繁的, 则候选 $\{P, Q, R\}$ 被剪枝。(这种剪枝在候选剪枝步骤进行。)

假设对表 7-14 所示的数据集实用 Apriori 算法。假定最小支持度阈值等于 30%, 即在少于 3 个事务中出现的任何项集都被视为非频繁的。

1) 绘制表示可能由表 7-14 产生的所有可能项集的格结构。

在得到的图中, 使用如下字母标记每个节点:

- **N**: 如果项集不被 Apriori 算法看作候选项集。一个项集不被看作候选项集有两种原因:
 - ① 在候选产生步骤未被产生。
 - ② 在候选产生步骤被产生, 但是因为发现它的一个子集是非频繁的而在其后的候选剪枝步骤被删除。
- **F**: 如果候选项集被 Apriori 算法发现是频繁的。
- **I**: 如果候选项集在支持度计数后被发现是非频繁的。

2) 被 Apriori 算法找到的频繁项集(相对于格中所有项集)所占的百分比是多少?

3) Apriori 算法在该数据集上的剪枝率是多少?(剪枝率 de_ned 定义为被认为不是候选的项集所占的百分比, 因为在候选产生时不产生它们, 或者在候选剪枝步被剪裁。)

4) 假报警率(即进行支持度计数之后, 被发现是非频繁的候选项集所占的百分比)是多少?

3. 给定图 7-2 所示的格结构和表 7-14 中的事务, 用如下字母标记每个节点:

- **M**: 如果它是极大频繁项集[⊖]。
- **C**: 如果它是频繁闭项集[⊖]。
- **N**: 如果它是频繁的, 但既不是极大的, 也不是闭的。
- **I**: 如果它是非频繁的。

假定最小支持度阈值等于 30%。

4. 考虑兴趣度量

$$m = \frac{P(B|A) - P(B)}{1 - P(B)}$$

对于关联规则 $A \Rightarrow B$,

1) 该度量的值域是什么? 何时该度量达到它的最大值和最小值?

⊖ 极大频繁项集是频繁项集, 并且它的任何超集(包含它的项集)都不是频繁的。——译者注

⊖ 频繁闭项集是频繁项集, 并且它的任何超集的支持度都不等于它的支持度。——译者注

表 7-14 购物篮事务的例子

事务 ID	购买的商品
1	{A, B, D, E}
2	{B, C, D}
3	{A, B, D, E}
4	{A, C, D, E}
5	{B, C, D, E}
6	{B, D, E}
7	{C, D}
8	{A, B, C}
9	{A, D, E}
10	{B, D}

- 2) 当 $P(A, B)$ 增加, 而 $P(A)$ 和 $P(B)$ 不变时, m 如何变化?
 - 3) 当 $P(A)$ 增加, 而 $P(A, B)$ 和 $P(B)$ 不变时, m 如何变化?
 - 4) 当 $P(B)$ 增加, 而 $P(A, B)$ 和 $P(A)$ 不变时, m 如何变化?
 - 5) 该度量在变量交换下是否对称?
 - 6) 当 A 和 B 统计独立时, 该度量的值是什么?
 - 7) 在行或列按比例缩放运算下, 该度量是否保持不变?
 - 8) 在反演运算下, 该度量是否保持不变?
5. 给定表 7-15 中的医疗数据库, 其中包含婴儿和成人猝死(SD)案例。分析小儿麻痹疫苗(PV)注射是否对猝死有影响。

表 7-15 假想的医疗数据库

婴儿	成人	PV	SD	支持度计数
1	0	1	1	26
1	0	1	0	24
1	0	0	1	49
1	0	0	0	1
0	1	1	1	20
0	1	1	0	130
0	1	0	1	240
0	1	0	0	510

- 1) 计算规则 $PV \rightarrow SD$ 的支持度、置信度和兴趣度。
- 2) 直观地, 你认为规则 $PV \rightarrow SD$ 表明了什麼? 这与上一个问题的结果一致吗?
- 3) 分别对婴儿和成人计算规则 $PV \rightarrow SD$ 的支持度、置信度和兴趣度。
- 4) 你能够从上一个问题的结果得到什么有趣的结论?

参考文献

- [1] Rakesh Agrawal and Ramakrishnan Srikant, Fast Algorithms for Mining Association Rules, *Proceedings of the 20th VLDB Conference*, Chile 1994.
- [2] Nitin R. Patel and Peter C. Bruce, *Data Mining In Excel: Lecture Notes and Cases*, Quantlink Corp., 2004.

第8章 用开源和商业软件进行机器学习

8.1 用 Weka 进行机器学习

Weka[1]是一个功能全面的机器学习和数据挖掘应用程序平台。Weka 是新西兰怀卡托 (Waikato) 大学开发的。“Weka”表示 Waikato Environment for Knowledge Analysis (知识分析怀卡托环境)。(Weka(新西兰黑秧鸡)也是一种具有好奇天性的不会飞的鸟,只能在新西兰的岛屿上看到。)Weka 用 Java 编程,并在所有操作系统下广泛测试。

在本章中,我们的目标是向读者介绍图形用户界面下的 Weka 的使用方法。这些内容基于 Weka 3-4-3。由于 Weka 的类层次结构进行了重组,例子(在下次重组前)只能用版本 3-4-3 运行。基本概念和问题可以转换到早期的版本上,但是具体的例子需要稍加改写。尽管对于最初的实验图形用户界面足够了,但是对于深层使用,建议使用命令行界面,因为它提供了一些通过图形用户界面无法使用的功能,并且使用更少的内存。

Weka 提供了优秀的学习算法,可以用于数据集中。它还包括各种数据集变换工具,如离散化算法。我们可以对数据集进行预处理,将它提供给学习系统,并且分析结果分类器和它的性能。在前面几章中,我们已经从理论上学习了算法。现在将使用这些算法并做一些机器学习工作。

假设我们有一些数据,并且想用它建立决策树。数据通常存储在电子数据表或数据库中。然而,Weka 期望它是 ARFF 格式,因为掌握每个属性的类型信息是必要的,而这种信息不能自动地从属性值中推断出来。ARFF (Attribute-Relation File Format, 属性关系文件格式)文件是一种 ASCII 文本文件,描述一个共享属性集的实例的列表。ARFF 文件是为 Weka 机器学习软件开发的。因此,在你将任何算法用于数据之前,必须将数据转换成 ARFF 格式。这可以很容易做到。ARFF 文件的主要部分由所有实例的列表组成,每个实例的属性值用逗号分隔。大部分电子数据表和数据库程序都支持将数据作为记录的列表输出到逗号分隔格式的文件中,其中项用逗号分开。一旦完成这项工作,只需要将该文件装入文本编辑器或字处理器;使用@relation tag 加上数据集的名字,使用@attribute 加上属性信息,再加上一个@data 行;将该文件作为原始文本文件保存,就完成了转换!

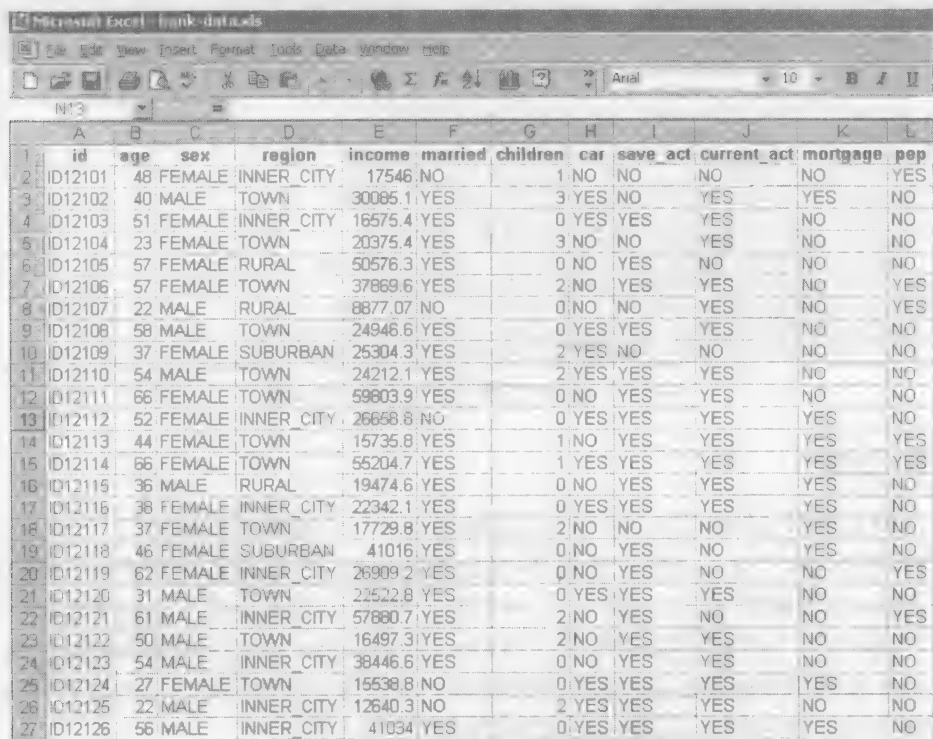
现在,让我们通过一些例子执行 Weka 的一些数据预处理和挖掘操作。Weka 软件可在本书附带的光盘上找到。安装 Weka 相当容易。这个例子使用的样本数据集是银行数据,数据采用逗号分隔格式(bank-data.csv),在本书附带的光盘上可以找到。光盘上还有其他一些标准数据集。建议读者使用这些数据集进行学习。

由于有些人很难克服使用这种软件的最初的惯性,我们在随书光盘上包括了一些动画,介绍如何对数据集 bank-data.csv 进行预处理、离散化、关联规则挖掘、分类和聚类。

8.1.1 开始

我们的数据(bank-data.csv)[2]以逗号分隔格式存放。但是在大多数情况下,数据存放在微软的 Excel 电子数据表中,我们把这些数据以逗号分隔的格式存放。打开 Excel 文件;首先

从 File(“文件”)下拉菜单选择 Save AS(“另存为”)。然后,在确认对话框选择 CSV,并保存该文件。图 8-1 以 Excel 格式显示我们的实例数据集。图 8-2 显示如何将 xls 格式转换成 csv 格式。



	A	B	C	D	E	F	G	H	I	J	K	L
	id	age	sex	region	income	married	children	car	save_act	current_act	mortgage	pep
2	ID12101	48	FEMALE	INNER_CITY	17546	NO		1 NO	NO	NO		YES
3	ID12102	40	MALE	TOWN	30065.1	YES		3 YES	NO	YES	YES	NO
4	ID12103	51	FEMALE	INNER_CITY	16575.4	YES		0 YES	YES	YES	NO	NO
5	ID12104	23	FEMALE	TOWN	20375.4	YES		3 NO	NO	YES	NO	NO
6	ID12105	57	FEMALE	RURAL	50576.3	YES		0 NO	YES	NO	NO	NO
7	ID12106	57	FEMALE	TOWN	37869.6	YES		2 NO	YES	YES	NO	YES
8	ID12107	22	MALE	RURAL	8877.07	NO		0 NO	NO	YES	NO	YES
9	ID12108	58	MALE	TOWN	24946.6	YES		0 YES	YES	YES	NO	NO
10	ID12109	37	FEMALE	SUBURBAN	25304.3	YES		2 YES	NO	NO	NO	NO
11	ID12110	54	MALE	TOWN	24212.1	YES		2 YES	YES	YES	NO	NO
12	ID12111	66	FEMALE	TOWN	59603.9	YES		0 NO	YES	YES	NO	NO
13	ID12112	52	FEMALE	INNER_CITY	26658.8	NO		0 YES	YES	YES	YES	NO
14	ID12113	44	FEMALE	TOWN	15735.8	YES		1 NO	YES	YES	YES	YES
15	ID12114	66	FEMALE	TOWN	55204.7	YES		1 YES	YES	YES	YES	YES
16	ID12115	36	MALE	RURAL	19474.6	YES		0 NO	YES	YES	YES	NO
17	ID12116	38	FEMALE	INNER_CITY	22342.1	YES		0 YES	YES	YES	YES	NO
18	ID12117	37	FEMALE	TOWN	17729.8	YES		2 NO	NO	NO	YES	NO
19	ID12118	46	FEMALE	SUBURBAN	41016	YES		0 NO	YES	NO	YES	NO
20	ID12119	62	FEMALE	INNER_CITY	26909.2	YES		0 NO	YES	NO	NO	YES
21	ID12120	31	MALE	TOWN	22522.8	YES		0 YES	YES	YES	NO	NO
22	ID12121	61	MALE	INNER_CITY	57880.7	YES		2 NO	YES	NO	NO	YES
23	ID12122	50	MALE	TOWN	16497.3	YES		2 NO	YES	YES	NO	NO
24	ID12123	54	MALE	INNER_CITY	38446.6	YES		0 NO	YES	YES	NO	NO
25	ID12124	27	FEMALE	TOWN	15538.8	NO		0 YES	YES	YES	YES	NO
26	ID12125	22	MALE	INNER_CITY	12640.3	NO		2 YES	YES	YES	NO	NO
27	ID12126	56	MALE	INNER_CITY	41034	YES		0 YES	YES	YES	YES	NO

图 8-1. Excel 格式数据集

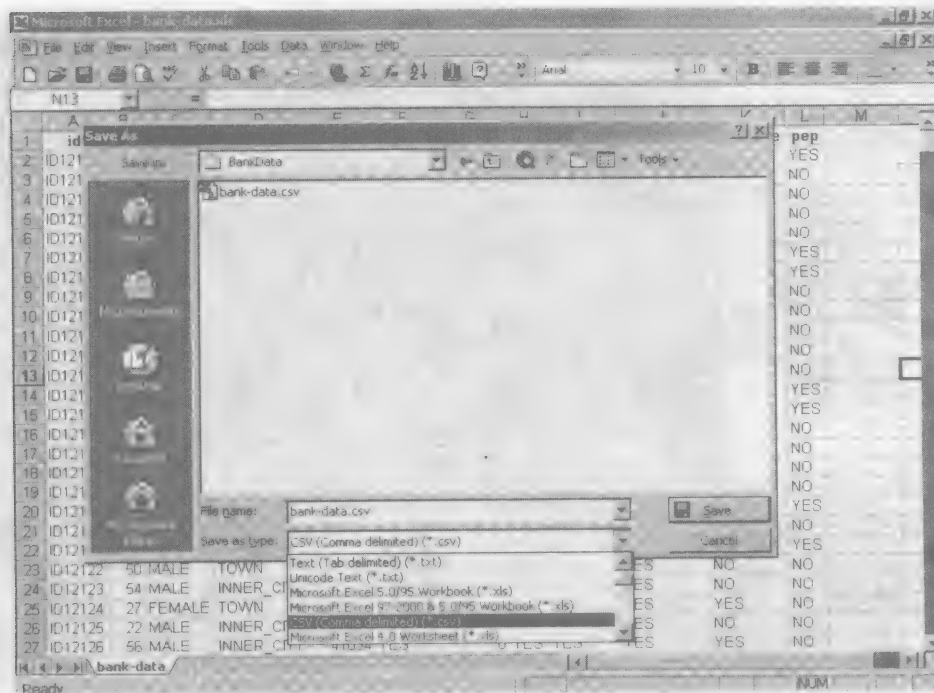


图 8-2 将 xls 格式转换成 csv 格式

该 csv 文件可以在文本编辑器中打开, 如图 8-3 所示。

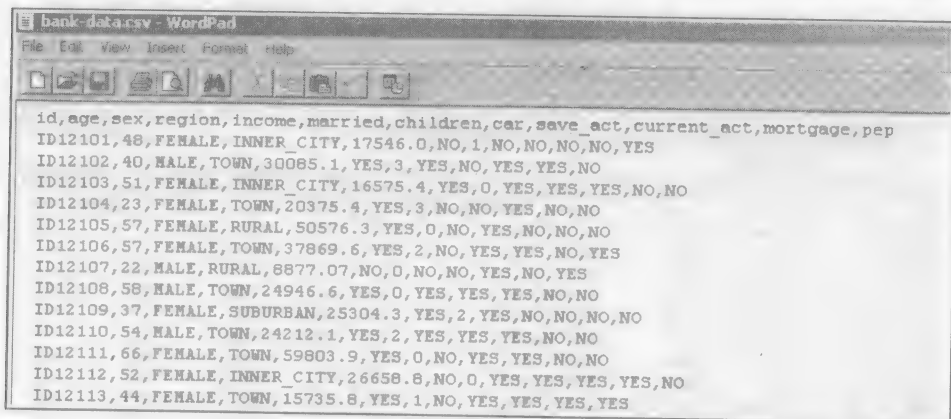


图 8-3 在文本编辑器中打开的 csv 文件

8.1.2 装入数据

除了 ARFF 数据文件格式之外, Weka 具有读入“.csv”格式文件的能力。正如可以在样本数据文件中看到的, 第一行包含属性名(用逗号隔开); 后面是每个数据行, 属性值以相同的次序列出(也用逗号隔开)。事实上, 一旦装入 Weka, 数据集就可以保存为 ARFF 格式。

在这个例子中, 我们将该数据集装入 Weka, 使用 Weka 的属性和离散化过滤执行一系列操作, 然后在结果数据集上进行关联规则挖掘。尽管所有操作都可以通过命令行执行, 但是我们使用 Weka Knowledge Explorer(知识探测器)的 GUI 界面。

假定 Weka 已经正确安装, 让我们装入 Weka。点击 explorer 命令框(参见图 8-4)。



图 8-4 Weka GUI 的开始面板

开始, (在 Preprocess tab) 点击“Open”, 并导航到包含数据文件(. csv 或 . arff) 的目录。在这个例子中, 我们将打开 bank-data.csv 文件, 如图 8-5 所示。



图 8-5 打开文件对话框

一旦装入数据, Weka 将识别属性, 并在数据扫描期间计算每个属性的一些基本统计量。图 8-6 左边的面板显示已识别出的属性的列表, 而上面的面板指出基本关系(表)和当前工作关系名(开始时, 它们相同)。

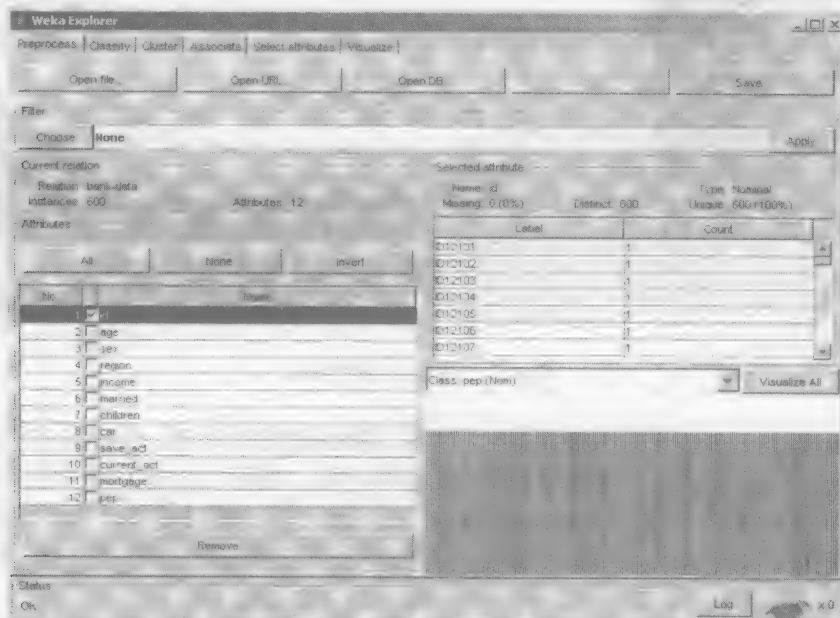


图 8-6 显示属性和它们的统计量的面板

点击左边面板中的任意属性将显示该属性上的基本统计量。对于分类属性，将显示每个属性值的频度，而对于连续属性，我们可以得到最小值、最大值、均值、标准差等。这显示在图 8-6 中。

8.1.3 选择或过滤属性

在我们的样本数据文件中，每个记录由顾客 id (“id”属性)唯一地标识。在数据挖掘之前，我们需要删除该属性。首先标记对应于属性 id(编号 1)检查框。点击左下方的 Remove 命令按钮。这将删除属性 id 和所有实例的 id 属性值。观察包含关于关系名、实例数和属性数的标记框。删除 id 属性后的属性数变成 11，它之前的值是 12。与 Weka 的早期版本相比，该操作变得非常容易。

记住，我们的数据仍然是 csv 格式。但是我们已经对 bank-data.csv 中原来的数据做了一些修改。为了将新的工作数据保存为 ARFF 文件，点击上方面板中的 save 按钮。这里，正如“save”对话框所示(见图 8-7)，我们将把新数据保存在文件“bank-data2.arff”中。



图 8-7 保存文件对话框

上述过程都属于对必须提供给机器学习算法的数据进行预处理。还有许多预处理选项可以使用。注意，可以选择多个过滤器并且同时使用它们。然而，在这个例子中，我们将一步步地应用不同的过滤器。现在还可以对新的工作关系使用其他过滤器。然而，在这个例子中，我们将把中间结果作为单独的数据文件保存，并且把每一步作为一次单独的 Weka 会话。

让我们在文本编辑器中打开保存的 bank-data2.arff 文件。图 8-8 显示了新产生的 ARFF

文件(在 WordPad 中)。注意,在新数据集中,“id”属性和记录中的所有对应值都已经删除。注意该文件并观察第一行@relation bank-data-weka.filters.unsupervised.attribute.Remove-R1。这个语句简单地描述迄今为止在数据集上做过的操作。属性可以是数值和标称类型。注意,属性 car 和 region 是标称属性。括号中给出了这些属性的可能取值。age、income 或 salary 这些属性是数值属性,它们可以取实数值。

```
@relation bank-data-weka.filters.unsupervised.attribute.Remove-R1

@attribute age numeric
@attribute sex (FEMALE,MALE)
@attribute region (INNER_CITY,TOWN,RURAL,SUBURBAN)
@attribute income numeric
@attribute married (NO,YES)
@attribute children numeric
@attribute car (NO,YES)
@attribute save_act (NO,YES)
@attribute current_act (NO,YES)
@attribute mortgage (NO,YES)
@attribute pep (YES,NO)

@data

48,FEMALE,INNER_CITY,17546,NO,1,NO,NO,NO,NO,YES
40,MALE,TOWN,30085.1,YES,3,YES,NO,YES,YES,NO
51,FEMALE,INNER_CITY,16575.4,YES,0,YES,YES,YES,NO,NO
23,FEMALE,TOWN,20375.4,YES,3,NO,NO,YES,NO,NO
57,FEMALE,RURAL,50576.3,YES,0,NO,YES,NO,NO,NO
57,FEMALE,TOWN,37869.6,YES,2,NO,YES,YES,NO,YES
22,MALE,RURAL,8877.07,NO,0,NO,NO,YES,NO,YES
58,MALE,TOWN,24946.6,YES,0,YES,YES,YES,NO,NO
37,FEMALE,SUBURBAN,25304.3,YES,2,YES,NO,NO,NO,NO
54,MALE,TOWN,24212.1,YES,2,YES,YES,YES,NO,NO
66,FEMALE,TOWN,59803.9,YES,0,NO,YES,YES,NO,NO
52,FEMALE,INNER_CITY,26658.6,NO,0,YES,YES,YES,YES,NO
44,FEMALE,TOWN,15735.8,YES,1,NO,YES,YES,YES,YES
66,FEMALE,TOWN,55204.7,YES,1,YES,YES,YES,YES,YES
36,MALE,RURAL,19474.6,YES,0,NO,YES,YES,YES,NO
38,FEMALE,INNER_CITY,22343.1,YES,0,YES,YES,YES,YES,NO
```

图 8-8 WordPad 中的 ARFF 文件

8.1.4 离散化

有些技术,如关联规则挖掘,只能在分类数据上进行。这要求在数值或连续属性上进行离散化。该数据集中有三个这样的属性:age、income 和 children。对于属性 children(数值),可能的值域只是 0、1、2 和 3。在这种情况下,我们选择保留数据中的这些值。这意味着我们可以通过删除作为属性 children 类型的关键字“numeric”,并用该值集取代它而进行离散化。我们直接在文本编辑器中完成这项工作,如图 8-9 所示。在 WordPad 中编辑完成后,保存该文件。可能有警告消息弹出。只需点击 OK 忽略该信息。

我们依靠 Weka 对属性 age 和 income 进行离散化。在这个例子中,我们把每个属性都划分成 3 个箱(区间)。Weka 的离散化过滤器可以探索地划分这些值域,或者使用各种统计学技术自动地确定划分数据的最佳方法。在这个例子中,我们将进行简单分箱。

首先,我们将通过打开文件 bank-data2.arff 将过滤后的数据集装入 Weka。这个“打开”对话框如图 8-10 所示。

现在让我们离散化新文件的属性。这次,我们激活 Filters 对话框(见图 8-11)。

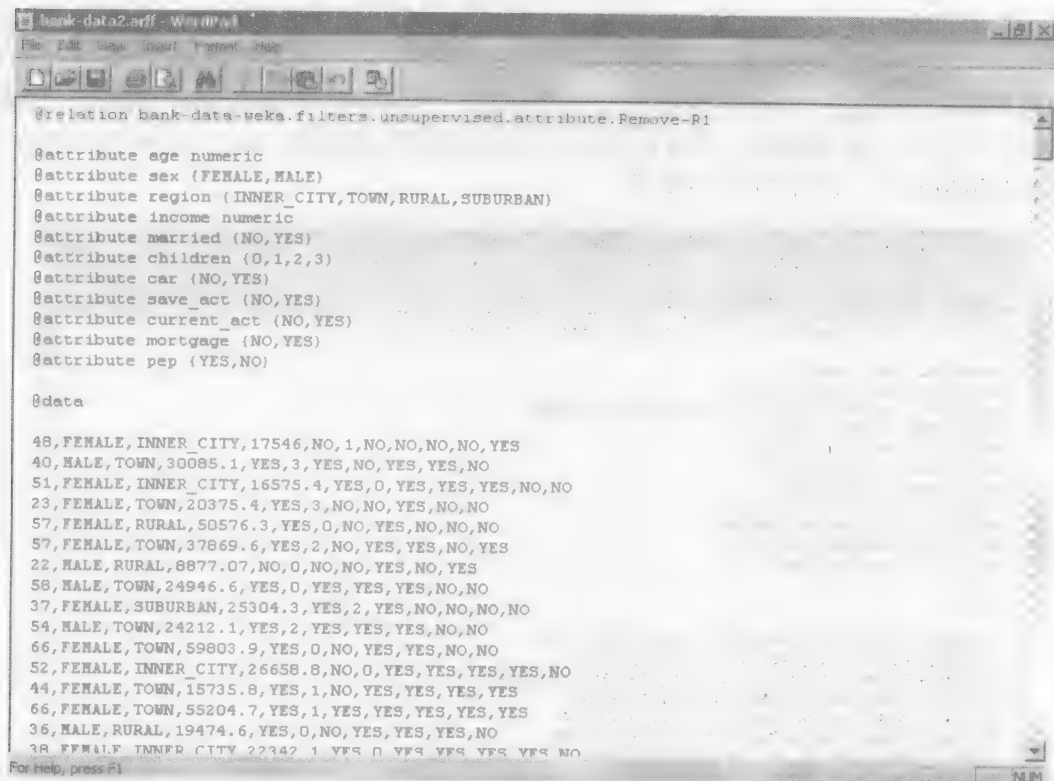


图 8-9 通过将 numeric 关键字改变为集合 {0, 1, 2, 3} 而对变量 children 离散化



图 8-10 打开文件对话框

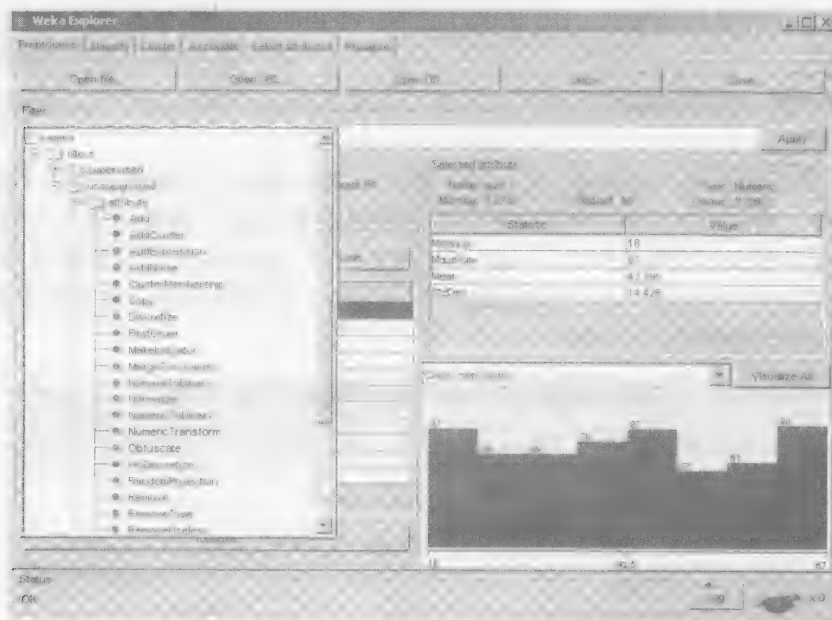


图 8-11 过滤器对话框

选择 weka.filters.unsupervised.attributes.Discretize。过滤器对话框中的文本框将出现像 Discretize-B 10-M-1.0-R first-last 这样的内容。点击该文本框将出现 DiscretizeFilter 对话框。

我们输入每个待离散化的属性的编号。在该例中，我们在对应于 AttributeIndices 的文本框输入 1, 4(对应于属性 age 和 income)。我们还输入 3 作为箱数(注意，如果对于不同的属性需要不同的箱数，我们必须为每个属性创建单独的过滤器)。由于我们进行简单分箱，因此，所有其他选项均设置为 false。该对话框如图 8-12 所示。

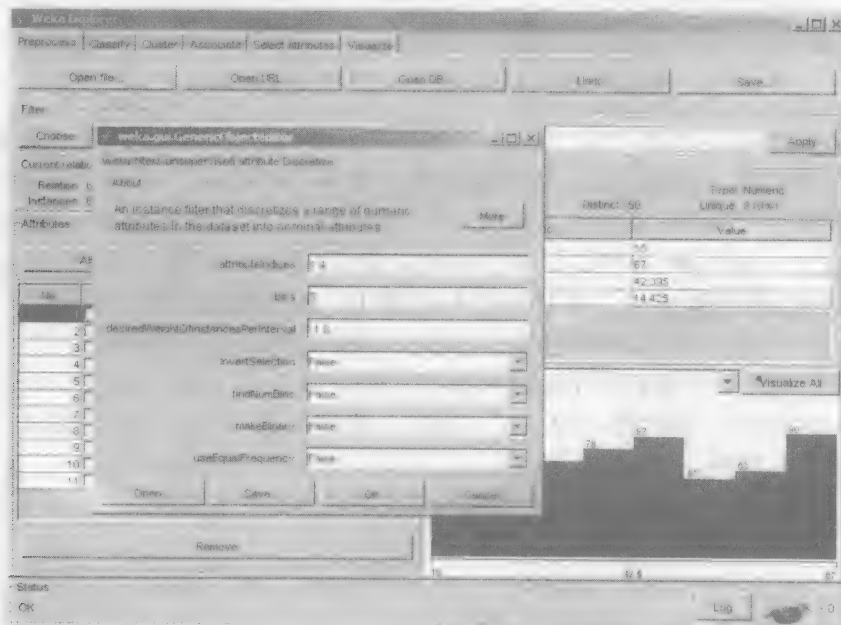


图 8-12 对连续变量分箱

这一过程将创建我们所需要的离散化过滤器。点击该对话框中的 OK。然后，在 Filter (过滤器) 面板点击 Apply。这将产生新的工作关系，其中两个被选中的属性都被划分成 3 个箱。为了检查结果，我们把这个新工作关系保存在文件 bank-data3.arff 中。如果有疑问，请读者观看附带光盘。

现在，让我们使用文本编辑器(在本例中是 WordPad)考察这个新数据集。数据的顶部显示在图 8-13 中。我们可以看到，Weka 已经将它自己的标记赋予离散化属性的每个值区间。例如，age 属性的较低区间标记为“(-inf - 34.333333]”(包括在单引号和转义字符中)，而中间的区间标记为“(34.333333 - 50.666667]”，等等。这些标记也出现在原来的数据记录中，表示对应的年龄值所在的区间。

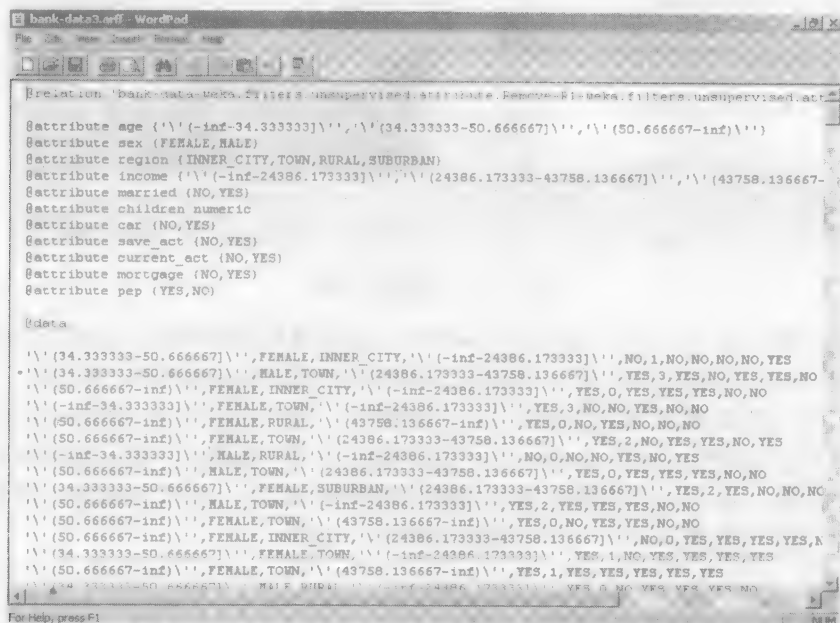


图 8-13 使用文本编辑器改变数值变量的界限

显然，Weka 的标记尽管可读，但是与命名习惯相去甚远。因此，我们将使用 WordPad 的搜索/替换功能，用更简洁、更可读的标记替换这些标记。幸而，WordPad 具有功能强大的正则表达式模式匹配能力，从而支持我们做这件事。图 8-14 显示用标记“0_34”替换年龄标记“(-inf - 34.333333]”的 WordPad 搜索/替换对话框。注意，regular expression (正则表达式) 选项被选中。在“Find what”框中，我们输入了整个标记“\ '(-inf - 34.333333] \ '”(包括反斜杠和单引号)。此外，反斜杠用另一个反斜杠转义，使得在正则表达式模式匹配中，将它们按字面意义处理(导致“\ \ '(-inf - 34.333333] \ \ ’”)。在“Replace with”框中，我们键入“0_34”。现在，我们点击“Replace All”按钮，用新模式替换旧模式的所有实例。

类似地，我们将使用相同的技术，用新的字符串替换如下字符串。

属性 age

- \ '(-inf (34.333333 \ "→0_34
- \ '(34.333333 - 50.666667 \ "→35_51
- \ '(50.666667 - inf \ "→52_max

属性 income

- ' \ ' (inf(24386.173333 \ "→0_24386
- ' \ ' 24386.173333 - 43758.173333 \ "→24387_43758
- ' \ ' 43578.173333 - inf \ "→43759_max

现在, 我们再将该 ARFF 文件中的关系名改为“bank-data-final”, 并将该文件保存为“bank-data-final.arff”(见图 8-15)。

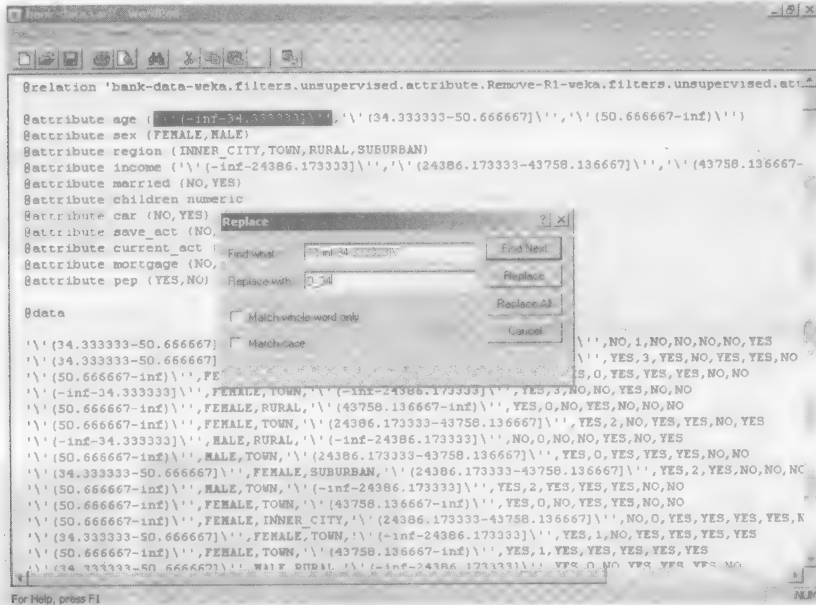


图 8-14 搜索和替换对话框



图 8-15 搜索和替换后的结果

8.1.5 关联规则挖掘

现在,我们已经准备就绪,进入数据挖掘步骤,并在我们的数据集上发现关联规则。图8-16显示打开新的数据文件“bank-data-final.arff”后的 Weka Explorer 界面。注意,离散化变量显示在窗口中。

不同的关联规则表达数据集的不同的规律性,并且它们通常预测不同的事情。



图 8-16 打开文件后的对话框

单击“Associate”将显示出“Apriori”算法的界面。如果你点击这个选项卡,你将会看到 Apriori 算法是所提供的三个关联规则算法中的一个。在“Associator”文本框(显示默认命令行)中点击后,出现 Apriori 对话框。该对话框如图 8-17 所示。这里,我们可以指定与 Apriori 相关的各种参数。点击“More”按钮,观察不同参数的一览表。

Weka 支持结果规则按照不同的度量标准(如置信度、挺度和提升度)排序。在这个例子中,我们选择提升度作为标准。此外,我们输入了 1.5 作为提升度(或改进)的最小值。规则的提升度用规则的置信度除以规则右端(RHS)项集的支持度计算。在一种简化形式下,给定一个规则 $L \Rightarrow R$,提升度(left)是 L 和 R 同时出现的概率与 L 和 R 的单独概率乘积的比,即

$$\text{left}(L \Rightarrow R) = \frac{\text{Pr}(L, R)}{\text{Pr}(L)\text{Pr}(R)}$$

如果该值为 1,则 L 和 R 是独立的。该值越高, L 和 R 在事务中一起出现就越不可能是随机出现,因为它们之间存在某种联系。

这里,我们还将规则的默认值(10)改为 100。最小支持度的上界设置为 1.0(100%),下界设置为 0.1(10%)。Weka 的 Apriori 从支持度上界开始,(以 Δ 增量,其默认值为 0.05 或 5%)逐渐降低支持度。当产生的规则达到指定个数或达到最小支持度下界时,算法停止。显著性检验选项仅用于置信度,并且默认值(-1.0)不使用。

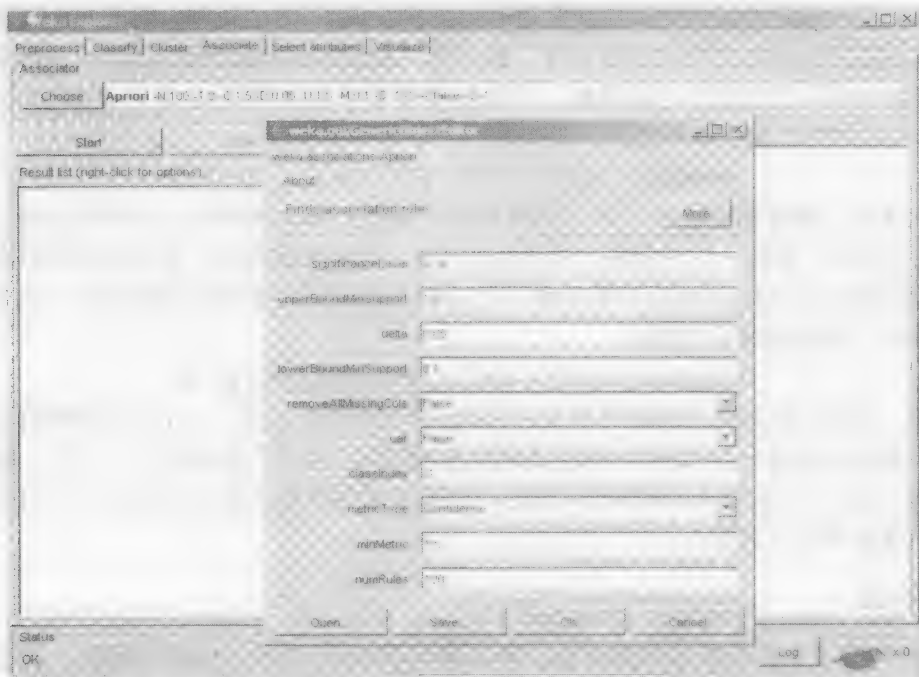


图 8-17 关联规则挖掘对话框

一旦设置好参数，“Associator”文本框将显示新的命令行。现在我们点击 Start 按钮，运行该程序。这会产生图 8-18 所示的规则集。我们把输出保存在 bank-data-ar1.txt 中。

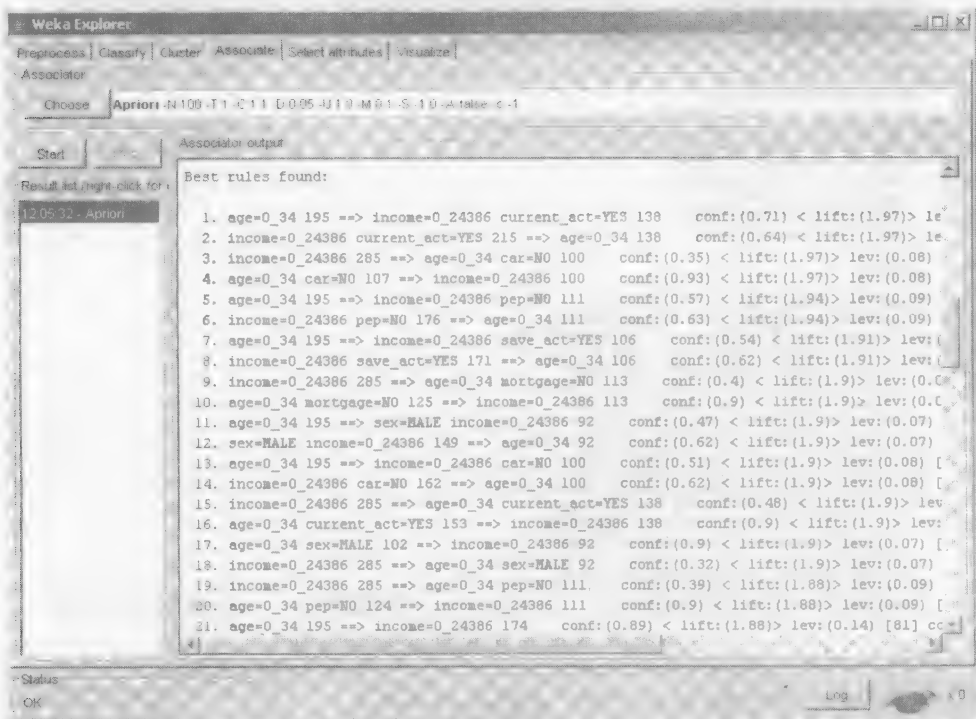


图 8-18 关联规则挖掘结果

注意, 基于指定的支持度和提升度阈值, 总共发现了 52 个规则。对于每个规则, 给出规则左部和右部的频度计数以及置信度、提升度、挺度 (leverage) 和确信度 (conviction) 值。注意, 挺度和提升度度量的内容类似, 只是挺度度量 L 和 R 同时出现的概率 (见以前的例子) 与 L 和 R 的每个独立概率之差, 即

$$\text{leverage}(L \Rightarrow R) = \text{Pr}(L, R) - \text{Pr}(L) \cdot \text{Pr}(R)$$

换句话说, 挺度度量被 L 和 R 二者覆盖的、超出 L 和 R 相互独立时所期望覆盖的附加案例所占的比例。这样, 对于挺度, 期望值大于 0, 而对于提升度, 我们希望看到值大于 1。最后, 确信度 (conviction) 类似于提升度, 但是它度量规则右部不为真的影响。它也转换成比率。因此, 确信度用下式度量:

$$\text{conviction}(L \Rightarrow R) = \text{Pr}(L) \cdot \text{Pr}(\text{not } R) / \text{Pr}(L, R)$$

这样, 与提升度相反, 确信度不是对称的 (并且也没有上界)。在大部分情况下, 关注支持度、置信度和提升度或挺率来定量地度量规则的“质量”就足够了。然而, 就有效性 (usefulness) 和可行动性 (actionability) 而论, 规则的实际价值评价是主观的, 并且高度依赖于特定的领域和商务目标。

8.1.6 分类

Weka 实现了大量分类和预测算法。使用这些算法的基本思想是类似的。在这个例子中, 我们将使用银行数据的修改版本, 对新的实例使用 C4.5 算法进行分类 (注意, C4.5 在 Weka 中作为分类方法类 weka.classifiers.j48.J48 实现)。银行数据的修改 (较小) 版本可以在文件 bank.arff 中找到, 如图 8-19 所示。



图 8-19 WordPad 中的 bank.arff 文件

我们仍然从装入数据到 Weka 开始, 如图 8-20 所示。

下一步, 我们选择“Classify”(分类)选项卡。大约有 6 组算法, 我们选择文件夹

“trees”。由 trees，我们选择 J48 分类方法，如图 8-21 所示。注意，C4.5 是由 ID3 进化而来的，但与 ID3 不同，J48(C4.5 算法的 Java 实现)并不要求数值属性离散化。



图 8-20 打开文件对话框



图 8-21 选择分类方法

现在，我们可以在弹出的窗口中指定算法的各种参数。在这个例子中，我们接受默认值，但 `reduceErrorPruning` 被设置为 `True`。该参数的默认版本并不使用子树提升方法进行剪枝。选定的参数如图 8-22 所示。

在主面板的“Test option”(检验选项)下，我们选择 10 折交叉确认作为我们的评估方法。由于我们没有独立的检验数据集，为了对所生成模型的准确率进行合理评估，这是必要的。现在点击“Start”来产生模型。当模型构造完成后，树的 ASCII 码版本和评估统计量将出现

在右部的面板中。通过点击最新结果集(在左边面板的“Result list”中)并从弹出菜单中选择“View in separate window”, 我们可以在独立的窗口中观察这些信息。这些步骤和包含分类结果的结果窗口分别如图 8-23 和图 8-24 所示。

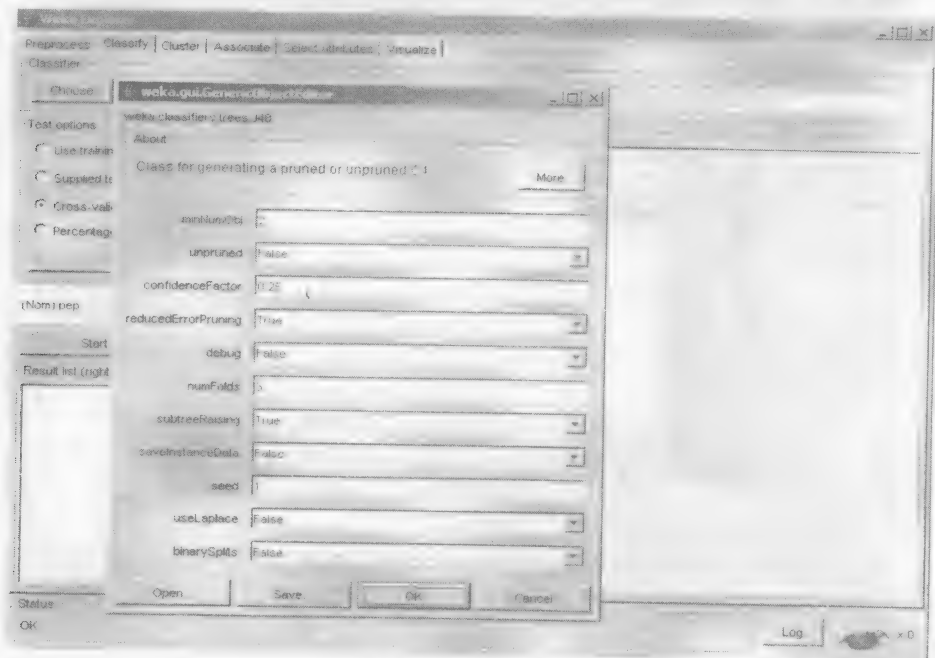


图 8-22 J48 对话框

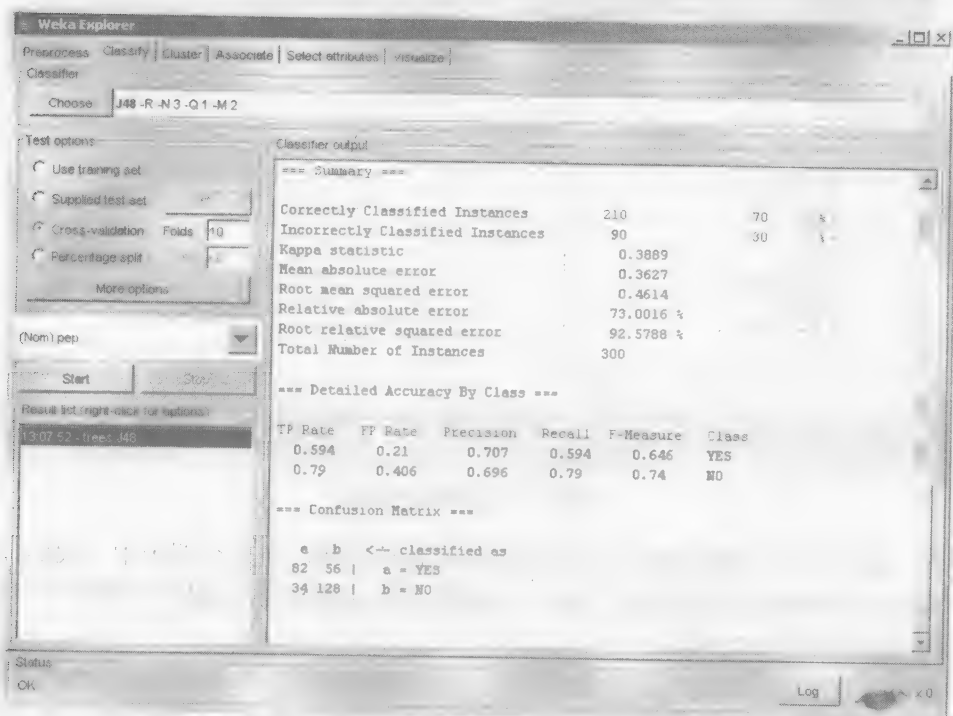


图 8-23 J48 决策树构造的结果

```

13:07:52 - trees.J48
=== Classifier model (full training set) ===

J48 pruned tree
-----

children = YES
| income <= 30099.3
| | car = YES: NO (35.0/9.0)
| | car = NO
| | | married = YES
| | | | mortgage = YES
| | | | income <= 13381: NO (2.0)
| | | | income > 13381: YES (9.0/2.0)
| | | | mortgage = NO: NO (16.0/7.0)
| | | married = NO: NO (19.0/6.0)
| income > 30099.3: YES (38.0/3.0)
children = NO
| married = YES: NO (58.0/14.0)
| married = NO
| | mortgage = YES: NO (9.0/1.0)
| | mortgage = NO: YES (14.0/1.0)

Number of Leaves :    9
Size of the tree :    17

Time taken to build model: 0.08 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances      210          70 %

```

图 8-24 使用 J48 的决策树

注意，我们的模型的分类准确率只有 70%（参见图 8-23）。这表明在构造另一个模型之前，我们可能需要做更多的工作（或者预处理，或者为分类选择正确的参数）。然而，在这个例子中，我们将继续使用该模型，尽管它不准确。

Weka 还允许我们观察分类树的图形形式。通过右击最新结果集（同上）并从弹出菜单选择“Visualize tree”就可以完成这一操作（见图 8-25）。注意，通过缩放窗口和选择树视图内的各种菜单项（使用鼠标右键），我们可以调整树视图，使它更可读（见图 8-26）。

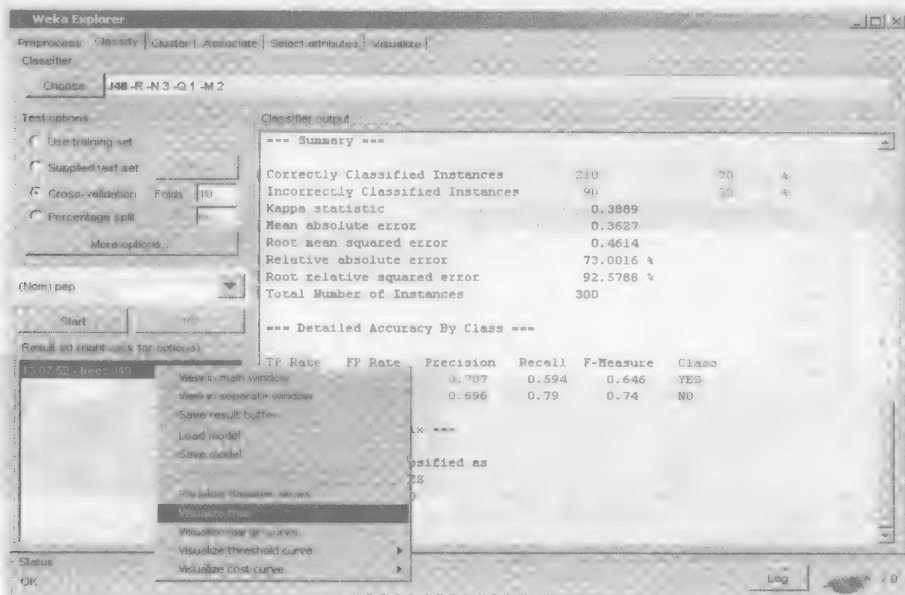


图 8-25 决策树可视化

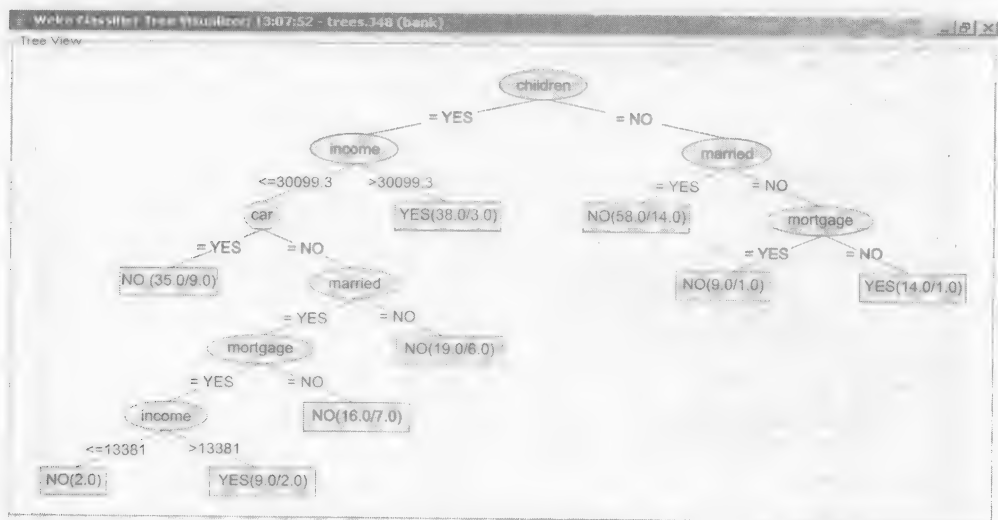


图 8-26 决策树

8.1.7 聚类

为了说明在 Weka 中进行聚类的过程，我们将使用 k-均值算法，对我们的银行数据集进行聚类。该数据集的这个版本的唯一变化是“children”属性已经从数值属性转换成分类属性。然而，对于聚类而言，这不是必需的。数据文件是 bank.arff，它包含 600 个实例。图 8-27 显示该文件装入后 Weka Explorer 的主界面。



图 8-27 主对话框

某些 k-均值算法的实现只支持数值属性。在这种情况下，有必要把数据集转换成标准的电子数据表，并将分类属性转换成二元属性。还有必要把依据不同尺度度量的属性（如“年龄”和“收入”）值规范化。尽管 Weka 提供了过滤程序来完成这些预处理任务，但是对于 Weka 中聚类，它们不是必需的。这是因为 Weka SimpleKMeans 算法自动处理分类和数值属性的混合。此外，在进行距离计算时，该算法自动规范化数值属性。Weka SimpleKMeans 算法使用欧氏距离计算实例和簇之间的距离。

为了进行聚类，选择该 Explorer 中的“Cluster”选项卡，并点击框中“Clusterers”标签，这时会出现可用聚类算法的下拉列表。在这个例子，我们选择“SimpleKMeans”，于是出现一个弹出窗口（如图 8-28 所示）。

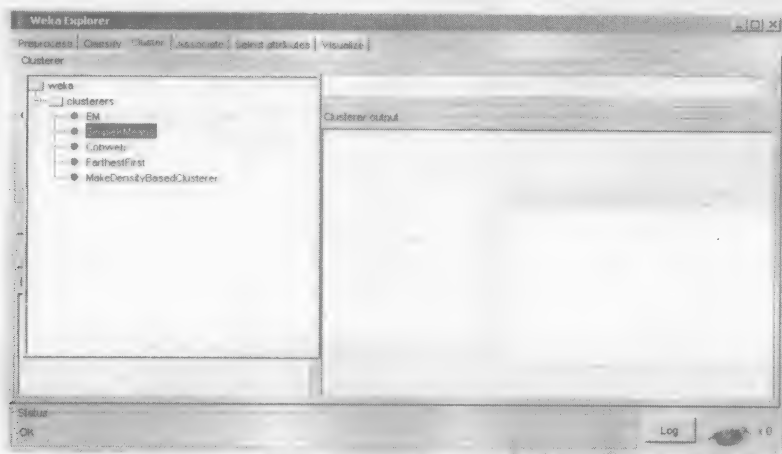


图 8-28 选择聚类算法

在图 8-29 所示的弹出窗口中，我们输入 6 作为簇个数，并且保留“seed”（种子）值不变。种子值用来产生随机数，而随机数又用来产生实例到簇的初始指派。注意，一般来说，k-均值对簇的初始指派相当敏感。这样，通常需要尝试不同的值并评估结果。

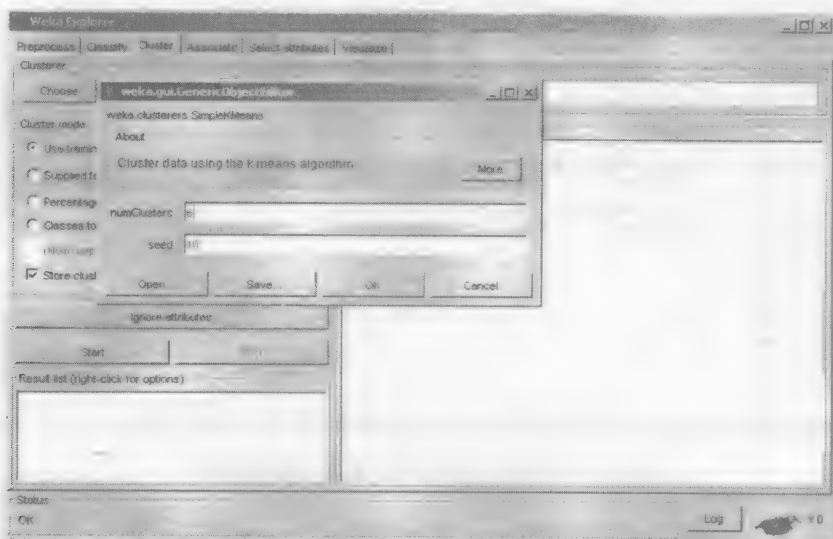


图 8-29 k-均值对话框

一旦选定选项，我们就可以运行该聚类算法。这里，我们确认在“Cluster Mode”面板中，已经选中“Use training set”选项，并且点击“Start”。与分类例子一样，我们可以右击“Result list”面板中的结果集，并在独立的窗口中观察聚类结果。这一过程和结果窗口显示在图 8-30 和 8-31 中。

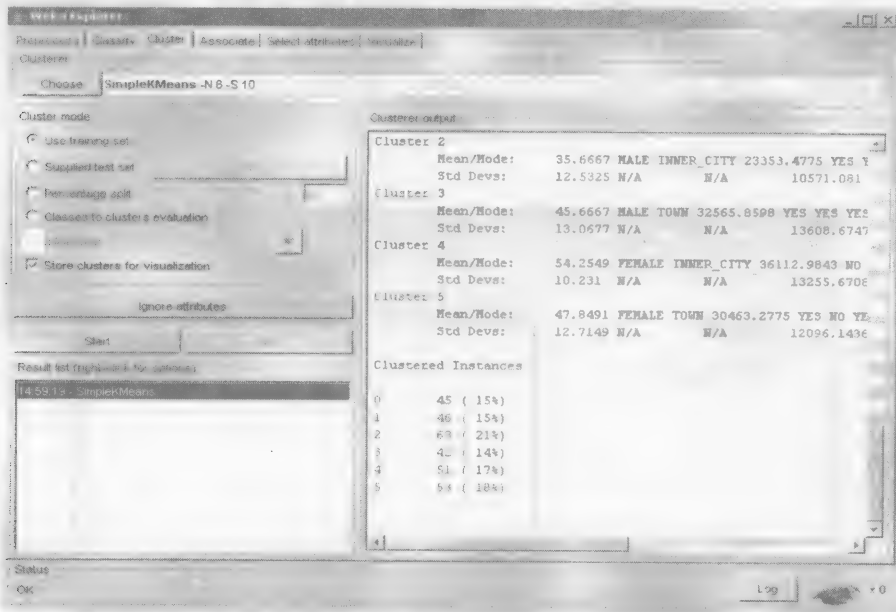


图 8-30 k-均值算法的结果

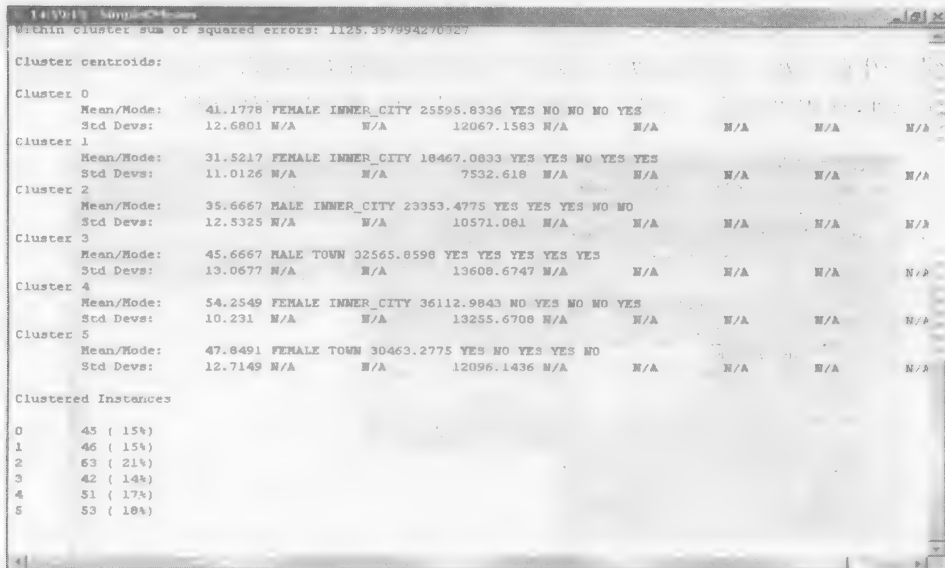


图 8-31 在独立的窗口中查看聚类算法的结果

结果窗口显示每个簇的质心，以及指派到不同簇的实例个数和百分比。簇的质心是每个簇的均值向量(因此，质心的每个维值表示簇中该维的均值)。这样，质心可以用来刻画簇。

例如，簇 1 的质心表示这是一个实例段，代表中年(大约 47 岁)女性，住在市区，平均收大约 \$32000，单身母亲(已婚 = yes；子女 = 2)。此外，这组人平均有汽车，有储蓄账号，但是没有抵押(并且平均具有 PEP(个人参股计划)产品)。

另一种理解每个簇的特征的方法是通过可视化。我们可以通过右击“Result list”面板中的结果集，并选择“Visualize cluster assignments”来实现。这时会弹出可视化窗口，如图 8-32 所示。



图 8-32 用不同的选项考查聚类结果

你可以为每个不同的维(x轴、y轴和颜色)选择簇号和其他任意属性。不同的选择组合将产生每个簇中不同联系的透视图。在前面的例子中，我们选择簇号作为x轴，(Weka设置的)实例数作为y轴，Sex(性别)属性作为颜色维。这使得每个簇中男性和女性分布的情况实现可视化。例如，你可以看到簇0和簇5被男性主宰，而簇3和簇4被女性主宰。在这种情况下，通过把颜色维改变成其他属性，我们可以看到它们在每个簇中的分布。

最后，我们可能愿意保存包括每个实例以及它们所指派的簇的结果数据集。为了完成这项工作，我们点击可视化窗口中的“Save”按钮，并将结果保存为文件“bank-kmeans.arff”。你可以使用所有菜单来浏览。

注意，除了“instance-number”属性外，Weka还在原数据集中添加了一个“Cluster”属性。在数据部分，现在每个实例都以它的指派簇作为最后一个属性值。在该数据集上执行一些简单的操作，就可以很容易地将它转换成更适合进行附加的分析和处理的数据集。例如，这里我们已经将这个数据集转换成逗号分隔的格式，并且按聚类的结果排序。此外，我们已经由任务2的原数据集(排序前)添加了ID字段。这些步骤的结果可以在文件“bank-kmeans.csv”中看到。

8.2 XLMINER

XLMINER 是适用于 Excel 的完整的数据挖掘附件。它是一种工具，可以用于快速开始数据挖掘，提供各种方法来分析数据。它广泛涵盖了用于分类、预测、亲合 (affinity) 分析、数据探查和归约的统计和机器学习技术。

XLMINER 的组件

分类	<ul style="list-style-type: none"> • 判别式分析 • Logistic 回归 • 分类树 • 朴素贝叶斯 • 神经网络 (多层前馈) • k-最近邻
预测	<ul style="list-style-type: none"> • 多元线性回归 • 回归树 • 神经网络 (多层前馈) • k-最近邻
亲合	<ul style="list-style-type: none"> • 关联规则
数据探查 和归约	<ul style="list-style-type: none"> • 主成分分析 • 层次聚类 • k-均值聚类

XLMINER 样本数据集

XLMINER 与“Datasets”文件夹中收集的样本数据集一起提供，这是 XLMINER 安装的基础目录。这些数据集是 Excel 文件。对于每个过程，XLMINER 用例子提供了广泛的联机帮助。样本数据集用于这些例子。

尽管某些数据集规模很小，但是对于解释数据挖掘技术，它们仍然是有用的。在许多情况下，大型数据集可以通过以下方法有效地处理：对数据集抽样建立模型，然后将从模型得到的结论用于整个数据集。

XLMINER 是商品化的软件，但是我们可以使用演示版本一个月。我们在本书光盘中提供了该软件的演示版本。请读者使用该软件内置的数据集进行实验。光盘上还包含如何使用该软件的说明。该软件还包含优秀的帮助文档，它是自解释的。

参考文献

- [1] Ian. H. Witten, *Data Mining Practical Machine Learning Tools and Techniques with Java Implementation*, Morgan Kautmann Publishers, 2000.
- [2] Banshad Mobasher, *Data Preparation and Mining with Weka—A Tutorial*. http://maya.cs.depaul.edu/~classes/ect_584/Weka/.

第9章 分类和回归算法

9.1 引言

在这一章中，我们将讨论一些经典但功能强大的方法。简单的方法通常行之有效，而且许多数据挖掘科学家在分析实际数据集的时候也推荐采用“简单优先”的方法论。

实际的数据集在很多时候具有以下的一些简单的结构和特征：

- 1) 一个简单属性解释了我们正在研究的现象，而其他属性是不相关的或是冗余的。
- 2) 所有属性都是相互独立的、对最终的输出的影响程度是相同的。
- 3) 如果一个基于几个变量的“if-then-else”逻辑结构即可描述事实，则这一逻辑结构可以很容易地用决策树来表示。
- 4) 几个独立的规则可以指导将各个实例分配到不同的类。
- 5) 变量的几个子集之间相互依赖。
- 6) 数值属性之间的线性依赖。
- 7) 实例本身之间的距离可以指导分类(详见第11章)。

在无限多种可能的数据集中，可能出现许多种不同的结构。某一类结构的数据挖掘工具，无论其多么强大，可能完全无法把握另一类的规律性，即使该类的结构很简单。其结果是不同的分类算法对相同的数据给出不同的结构。当一个分类算法输出一个晦涩难解的结构时，另一个算法却可能给出一个简单、直接而又全面的结构。

这就是“简单优先”的原则。而且，我们注意到一种分类方法可能无法适用于所有数据集。

现在我们讨论一个最简单的算法，称为“朴素贝叶斯”(Naïve Bayes)。

9.2 朴素贝叶斯

朴素贝叶斯模型假定所有变量对分类而言均是有用的，并且这些变量是相互独立的。换句话说，朴素贝叶斯模型假定所有变量是不相关的。对大多数数据集而言，这是一个不现实的假设。但是，这一假定在许多实际问题中可以产生一个简单的预测框架，并产生出乎意料的好结果。

为了说明这一框架，我们考虑在第4章中使用的气象数据(用于打网球)。表9-1是对气象数据的汇总，得到这些数据的方法是对play的每个值(yes或no)计数每个属性值对出现的次数。气象数据的汇总采用了便于进行朴素贝叶斯概率计算的格式。

表9-1 气象数据、计数和概率

天气			温度			湿度			有风			打网球	
yes no			yes no			yes no			yes no			yes	no
晴	2	3	热	2	2	高	3	4	否	6	2	9	5
多云	4	0	温暖	4	2	正常	6	1	是	3	3		

(续)

天气		温度		湿度		有风		打网球	
yes	no	yes	no	yes	no	yes	no	yes	no
雨	3	2	凉爽	3	1	否	6/9	9/14	5/14
晴	2/9	3/5	热	2/9	2/5				
多云	4/9	0/5	温暖	4/9	2/5				
雨	3/9	2/5	凉爽	3/9	1/5	是	3/9	3/5	

表 9-2 给出了新一天的属性值。朴素贝叶斯的目标是预测类别值(打网球 = yes 或打网球 = no)。朴素贝叶斯以概率的形式实现这一目标,即赋予“打网球 = yes”一个概率(也为打网球 = no 赋值)。

表 9-2 新一天的(气象)打网球

天气	气温	湿度	有风	Play
晴	凉爽	高	是	?

我们使用 $\Pr[A]$ 表示事件 A 发生的概率, $\Pr(A|B)$ 表示事件 A 在另一个事件 B 上的条件概率。假设 H 是比赛进行,即 play 为 yes,我们的目标是计算 $\Pr(H|E)$ 。 E 是证据。证据 E 就是新一天的气象数据属性值的某种组合:天气 = 晴,气温 = 凉爽,湿度 = 高,有风 = 是。我们相应地称它们为证据 E_1 、 E_2 、 E_3 、 E_4 。假定这些证据(在给定分类的前提下)是相互独立的,则它们的联合概率由相应的概率相乘获得,即

$$\Pr(\text{yes}|E) = \frac{\Pr(E_1|\text{yes})\Pr(E_2|\text{yes})\Pr(E_3|\text{yes})\Pr(E_4|\text{yes})\Pr(\text{yes})}{\Pr(E)} \quad (9.1)$$

上式最后的 $\Pr(\text{yes})$ 是在不知道任何证据 E (即不知道我们谈论的某一天的天气状况)的前提下“yes”的概率,称为假设 H 的先验概率(prior probability)。在我们的例子中,先验概率为 9/14,因为 14 个训练样本中有 9 个(play 属性的值)“yes”值。

式(9.1)中的分母可以不考虑:在最后的归一化(normalizing)步骤中通过使“yes”和“no”的概率之和为 1,我们可以消除它。

我们给式(9.1)中的分子一个新的名字——“打网球 = yes 的似然(likelihood)”。我们称之为似然的原因是其值反映了“打网球 = yes”出现的几率,但是它并非概率。

因此,“yes”的似然 = $2/9 \times 3/9 \times 3/9 \times 3/9 \times 9/14 = 0.0053$ 。

这些分数值取自表 9-1 中属性值为“yes”的列中和新一天的气象数据(见表 9-2)中的属性值相对应的值(概率值),而最后的 9/14 表示打网球为 yes 的天数所占的比例。使用类似的方法可以计算“打网球 = no”的似然 = $3/5 \times 1/5 \times 4/5 \times 3/5 \times 5/14 = 0.0206$ 。

这说明对(考察中的)新的一天而言,不打网球的可能性是打网球的可能性的四倍。通过归一化可以使得这些值转变为概率值使得其和为 1:

$$\text{Yes 的概率} = \frac{0.0053}{0.0053 + 0.0206} = 20.5\%$$

$$\text{No 的概率} = \frac{0.0206}{0.0053 + 0.0206} = 79.5\%$$

上述简单、直观的方法基于条件概率的贝叶斯定理。贝叶斯定理说明,如果存在一个假设 H 和基于这一假设的证据 E ,那么在我们的例子中

$$\Pr(H|E) = \frac{\Pr(E|H)\Pr(H)}{\Pr(E)} = \frac{\Pr(E_1|H)\Pr(E_2|H)\Pr(E_3|H)\Pr(E_4|H)\Pr(H)}{\Pr(E)} \quad (9.2)$$

其中 $E \equiv E_1 \cap E_2 \cap E_3 \cap E_4$ 。我们假定 E_1 、 E_2 、 E_3 、 E_4 是相互独立的，所以有：

$$\Pr(E|H) = \Pr(E_1|H)\Pr(E_2|H)\Pr(E_3|H)\Pr(E_4|H)$$

这一方法以朴素贝叶斯命名，因为它是基于贝叶斯定理的，并且简单地假定属性间的独立性——只有事件是相互独立的，其发生概率的连乘才是合法的。在现实中，属性间（在给定的类别）是相互独立的，显然是简单化的假定。尽管朴素贝叶斯的名字并不响亮，但是它在实际数据上测试时效果很好，特别是经过一些属性选择的过程，消除冗余，进而消除了一些非独立属性时尤其如此。

9.2.1 朴素贝叶斯的零频率问题

如果某个属性值不在每个类别值的训练集中都出现，则会出现问题。

例如，使用表 9-1 中的数据集，如果要预测表 9-3 中给定属性值的新的一天的类别值，则我们得到 $\Pr(\text{Yes}|E) = 1$ 和 $\Pr(\text{No}|E) = 0$ 。这是因为 $\Pr(\text{'天气' = '多云'} | \text{'打网球' = 'yes'}) = 1$ ，且 $\Pr(\text{'天气' = '多云'} | \text{'打网球' = 'no'}) = 0$ 。理论上这是正确的，因为只要天气是多云就能打网球。但是这个结果违背了朴素贝叶斯的基本假设：输出依赖于所有属性（依赖变量）。在目前这个例子中，输出只依赖于“天气 = 多云”。

表 9-3 新的一天的另一组数据

天气	气温	湿度	有风	打网球
多云	凉爽	高	是	?

这个问题通过稍稍改变根据频率计算概率的方法可以很容易地被解决。

例如，表 9-1 的上半部分显示，对打网球 = no 而言，天气是晴天的数据有三条，多云有零条，而雨天有两条。表的下半部分计算了三个事件的概率分别为 3/5，0/5，2/5。现在，我们可以为每个分子值加 1，然后相应的给分母值加 3，得到的概率分别为 4/8，1/8 和 3/8。这样可以保证出现零次的属性值得到一个小的、非零的概率值。为每个计数加一的策略是称为拉普拉斯估计 (Laplace estimator) 的标准技术，它以 18 世纪伟大的法国数学家皮埃尔·拉普拉斯 (Pierre Laplace) 的名字命名。尽管这一方法在实践中效果良好，但是没有充分的理由来为每个计数加 1。替代地，我们可以选择一个常数 μ 并使用下式计算：

$$\frac{3+\mu/3}{5+\mu}, \frac{0+\mu/3}{5+\mu}, \frac{2+\mu/3}{5+\mu}$$

现在上式是完全的贝叶斯公式，它为每个（事件的）概率计算都赋予了先验概率。这种方法是完全严格的，但缺点是应该如何进行先验概率赋值通常不是很明确。在实践中，除非有相当数量的训练数据实例，否则先验概率作用不大。因此人们通常使用拉普拉斯估计通过把所有的计数初始化为 1，而非 0，来估计频率。

9.2.2 缺失值和数值属性

贝叶斯方法的优点之一是数据缺失不会造成很大的问题。举例来说，如果在表 9-2 中没有天气趋势的数据值，那么在计算中只要简单地忽略掉这个属性，得到：

$$\text{yes 的似然} = 3/9 \times 3/9 \times 3/9 \times 9/14 = 0.0238$$

$$\text{no 的似然} = 1/5 \times 4/5 \times 3/5 \times 5/14 = 0.0343$$

这两个数值分别比之前的计算值高很多, 因为计算时少了一部分值。但是这么做是没有问题的, 因为这部分值在两种情况下都缺失了, 并且这些似然值容易归一化, 得到的 yes 和 no 的概率分别为 41% 和 59%。

如果一个训练实例中缺失了一个值, 那么在进行频率计数的时候只需忽略对其的计数, 同时在概率计算中使用实际出现的值的个数而非训练数据的总数。

对数值属性, 通常假定它们服从正态或高斯分布。表 9-4 给出了有数值属性的天气数据的汇总。(在数据的汇总中) 对标称属性而言, 计数仍然按照以前的方法进行, 而对数值属性只是列出所有出现的数值。然后, 对标称属性的计数被归一化为概率, 而每个类和每个数值属性计算均值和标准偏差。这样, “yes” 实例上的温度均值为 73, 其标准偏差是 6.2。

表 9-4 有数值属性的天气数据的汇总统计

天气			温度		湿度		有风			打网球			
	yes	no	yes	no	yes	no		yes	no	yes	no		
晴	2	3	83	85	86	85	无	6	2	9	5		
多云	4	0	70	80	96	90	有	3	3				
雨	3	2	68	65	80	70							
			64	72	65	95							
			69	71	70	91							
			75		80								
			75		70								
			72		90								
			81		75								
晴	2/9	3/5	均值	73	74.6	均值	79.1	86.2	无	6/9	2/5	9/14	5/14
多云	4/9	0/5	标准差	6.2	7.9	标准差	10.2	9.7	有	3/9	3/5		
雨	3/9	2/5											

对一个具有均值 μ 和标准偏差 σ 的正态分布而言, 其概率密度函数由下面的表达式给定:

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

上述描述意味着, 如果我们认为输出是 “yes”, 而此时有温度值, 比如说是 66, 那么我们只要在上面公式中插入 $x=66$, $\mu=73$ 以及 $\sigma=6.2$ 即可。所以概率密度函数的值是:

$$f(\text{温度}=66|\text{yes}) = \frac{1}{\sqrt{2\pi}6.2} e^{-\frac{(66-73)^2}{2 \times 6.2^2}} = 0.0340$$

使用相同的记号, 输出为 yes, 而湿度的值, 比如说是 90, 其概率密度可以以相同的方式计算如下:

$$f(\text{湿度}=90|\text{yes}) = 0.0221$$

在表 9-5 的新一天的数据中使用这些概率和密度函数值, 得到:

$$\text{yes 的似然} = 2/9 \times 0.0340 \times 0.0221 \times 3/9 \times 9/14 = 0.000036$$

$$\text{no 的似然} = 3/5 \times 0.0291 \times 0.0380 \times 3/5 \times 9/14 = 0.000136$$

通过上式可以得到下面的概率:

$$\text{yes 的概率} = \frac{0.000036}{0.000036 + 0.000136} = 20.9\%$$

$$\text{no 的概率} = \frac{0.000136}{0.000036 + 0.000136} = 79.1\%$$

这些值和先前使用表 9-2 的数据的计算结果非常接近, 因为气温值 66 和湿度值 90 发生的概率接近于前面我们使用的气温值为凉爽、湿度值为高发生的概率。

正态分布的假定使得很容易扩展朴素贝叶斯分类器来处理数值属性。如果任何数值属性值缺失, 那么我们只基于存在的值来计算均值和标准偏差。

表 9-5 另一个新的一天的天气数据

天气	气温	湿度	有风	打网球
多云	66	90	是	?

朴素贝叶斯以清晰的语义给出了表示、利用和学习概率知识的简单方法, 并且使用它可以获得很好的结果。在实践中, 已经多次证明朴素贝叶斯在许多数据集上不逊于甚至优于一些更复杂的分类方法。这里的原则是: 总是优先尝试简单的方法。经过不断尝试, 机器学习的研究者经过努力、试图使用更复杂的学习模型获得良好结果, 却在多年之后发现使用类似朴素贝叶斯的简单方法即可获得同样乃至更好的结果。

9.3 多元回归分析

9.3.1 什么是回归分析

回归分析是进行数据分析来解释关联和因果关系的统计方法。回归分析通常和相关性分析一起进行。相关性分析度量两组数量值之间的关联度, 而回归分析是基于一个变量或更多其他变量的变化来解释另一个变量的变化。其中, 被解释的变量称为因变量(dependent variable), 用于解释(因变量)变化的变量称为自变量(independent variable)。因此, 回归分析解释了存在于因变量和自变量之间的因果效应。

9.3.2 简单和多元回归分析

如果只有一个因变量, 并且只有一个自变量用于解释因变量的变化, 那么这个模型称为简单回归模型(simple regression model)。如果有多个自变量用于解释一个因变量的变化, 则这个模型称为多元回归模型(multiple regression model)。分析的过程称为多元回归分析(multiple regression analysis)。

9.3.3 在市场营销中的应用

回归分析在市场营销中的主要应用是根据若干自变量进行销售预测。回归分析还可以用来估计若干人口及心理因素之间的关系, 也可以用来确定单个满意度元素对总体满意度的相对影响。

9.3.4 方法

一般的回归模型具有下面的形式:

$$Y = a + b_1x_1 + b_2x_2 + \cdots + b_nx_n$$

其中 Y 是因变量, 而 x_1, x_2, \cdots, x_n 是期望和 Y 相关并可以用来解释和预测 Y 的自变量。 b_1, b_2, \cdots, b_n 是相应的自变量的系数, 这些系数可以通过输入数据来确定。

进行回归分析需要关于 Y 及每一个 x 变量的输入数据。输出由模型的所有自变量的 b 系数

组成。输出还给出模型中每个自变量的显著性的 t -检验结果, 以及整体模型的 F 检验结果。

9.3.5 使用 Excel 进行多元回归分析

为了使用微软的 Excel 进行多元回归分析, 我们首先需要将数据输入 Excel 工作表。因变量和自变量以矩阵形式输入 Excel 工作表。我们通过使用 Excel 解决一个实际生活中的问题来说明如何用 Excel 进行实现多元回归分析。这个问题描述如下: 一个电动马达的制造商和营销商希望构建一个有 6 个自变量的回归模型来预测销售, 并收集了来自 15 个区域的销售和 6 个自变量的历史数据。

我们假定数据来自公司的不同运营区域, 并且在这些区域收集的 6 个变量的数据如下:

- 因变量

Y = 该区域的销售额, 以万卢比为单位

- 自变量

x_1 = 该区域的市场潜力 (以万卢比为单位)

x_2 = 公司在该区域的经销商数量

x_3 = 该区域的销售人员数量

x_4 = 该区域竞争者活动的 5 级别评定 (竞争者的活动等级 1 = 低, 5 = 高)

x_5 = 该区域的服务人员数量

x_6 = 该区域已有的客户数量

这一问题的回归模型如下:

$$Y = a + b_1x_1 + b_2x_2 + b_3x_3 + b_4x_4 + b_5x_5 + b_6x_6 \quad (9.3)$$

9.3.6 输入数据

由 15 个观测值 (来自 15 个不同销售区域) 组成的数据集以矩阵形式输入 Excel。在数据输入后, 工作表如图 9-1 所示。

	Sales(y)	Potential(x1)	Dealers(x2)	People(x3)	Competitor(x4)	Service(x5)	Customer(x6)
1	5	25	1	6	5	2	20
2	60	150	12	30	4	5	50
3	20	45	5	15	3	2	25
4	11	30	2	10	3	2	20
5	45	75	12	20	2	4	30
6	6	10	3	8	2	3	16
7	15	29	5	18	4	5	30
8	22	43	7	16	3	6	40
9	29	70	4	15	2	5	39
10	3	40	1	6	5	2	5
11	16	40	4	11	4	2	17
12	8	25	2	9	3	3	10
13	18	32	7	14	3	4	31
14	23	73	10	10	4	3	43
15	81	150	15	35	4	7	70

图 9-1 微软 Excel 电子表格中用于回归的数据

在输入数据之后,如图 9-2 所示,点击 Tool 选择 Data analysis。

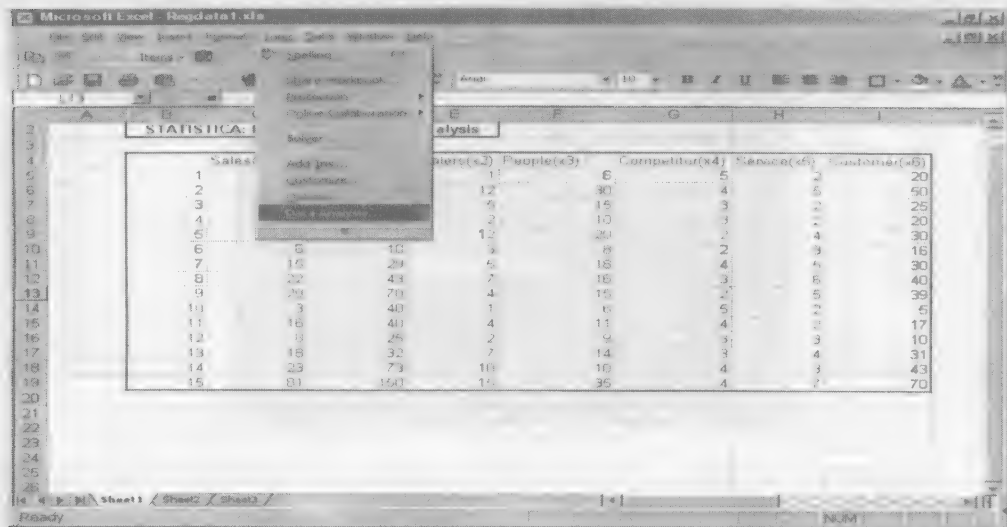


图 9-2 选择 Excel 中的数据分析师

在弹出的 Data analysis(数据分析)框中选择 Regression(回归),如图 9-3 所示。

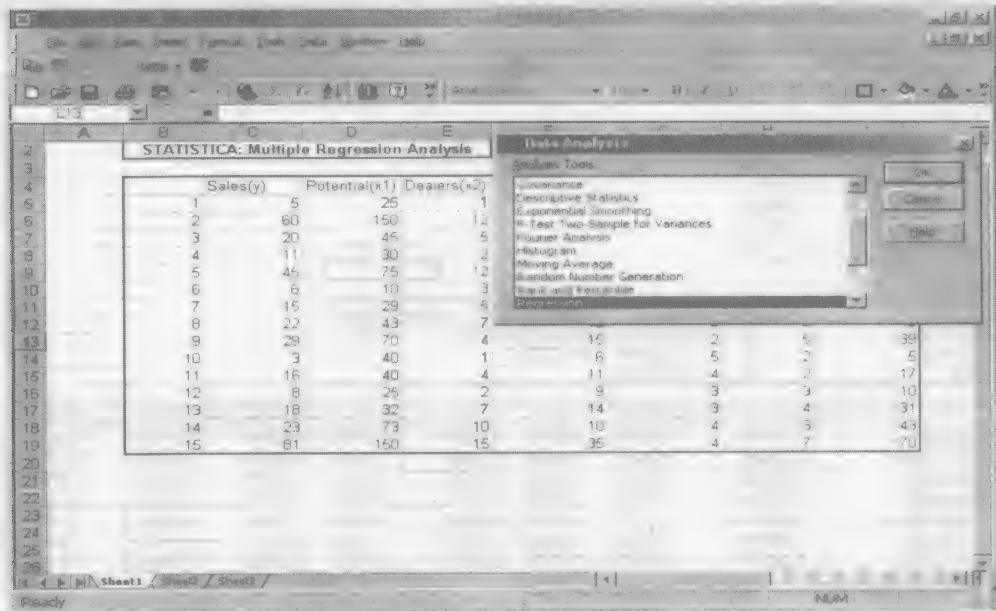


图 9-3 选择回归选项

在选择回归选项后,会弹出如图 9-4 所示的窗口。

在该窗口中分别为 Y 和 X 选定其所对应的因变量单元格区域和自变量单元格区域。然后进行分析,输出结果会出现在一个新的工作表中,如图 9-5 所示。

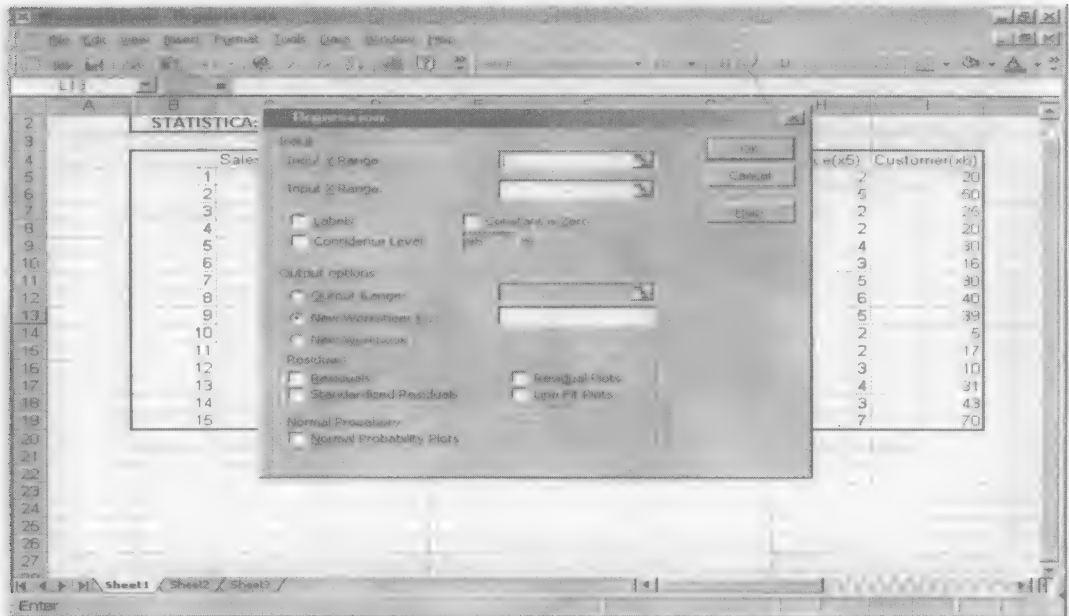


图 9-4 选定用于回归分析的输入和输出域

Microsoft Excel - Regdata1.xls

SUMMARY OUTPUT								
Regression Statistics								
Multiple R	0.98853							
R Square	0.97719							
Adjusted R Square	0.96009							
Standard Error	4.39102							
Observations	15							
ANOVA								
	df	SS	MS	F	Significance F			
Regression	6	6609.48459	1101.58077	57.13269	0.000004			
Residual	8	154.24874	19.28109					
Total	14	6763.73333						
	Coefficients	Standard Error	t Stat	P-value	Lower 95%	Upper 95%	Lower 95.0%	Upper 95.0%
Intercept	-3.17298	5.81339	-0.54581	0.60008	-16.57870	10.23274	-16.57870	10.23274
Potential(x1)	0.22685	0.07461	3.04044	0.01605	0.05480	0.39890	0.05480	0.39890
Dealers(x2)	0.81938	0.63127	1.29800	0.23046	-0.63632	2.27509	-0.63632	2.27509
People(x3)	1.09104	0.41812	2.60937	0.03116	0.12684	2.05523	0.12684	2.05523
Competitor(x4)	-1.89270	1.33971	-1.41276	0.19543	-4.98208	1.19669	-4.98208	1.19669
Service(x5)	-0.54925	1.56823	-0.35024	0.73520	-4.16560	3.06710	-4.16560	3.06710
Custom(x6)	0.06594	0.19500	0.33817	0.74394	-0.36373	0.51562	-0.36373	0.51562

图 9-5 Excel 的回归分析输出

9.3.7 回归输出

回归(分析)输出列出了自变量的系数。在图 9-5 中可以看到其位于 B 列的 17~23 行,

如下所示：

$$a(\text{截距(intercept)}) = -3.17298$$

$$b_1 = 0.22685$$

$$b_2 = 0.81938$$

$$b_3 = 1.09104$$

$$b_4 = -1.89270$$

$$b_5 = -0.54925$$

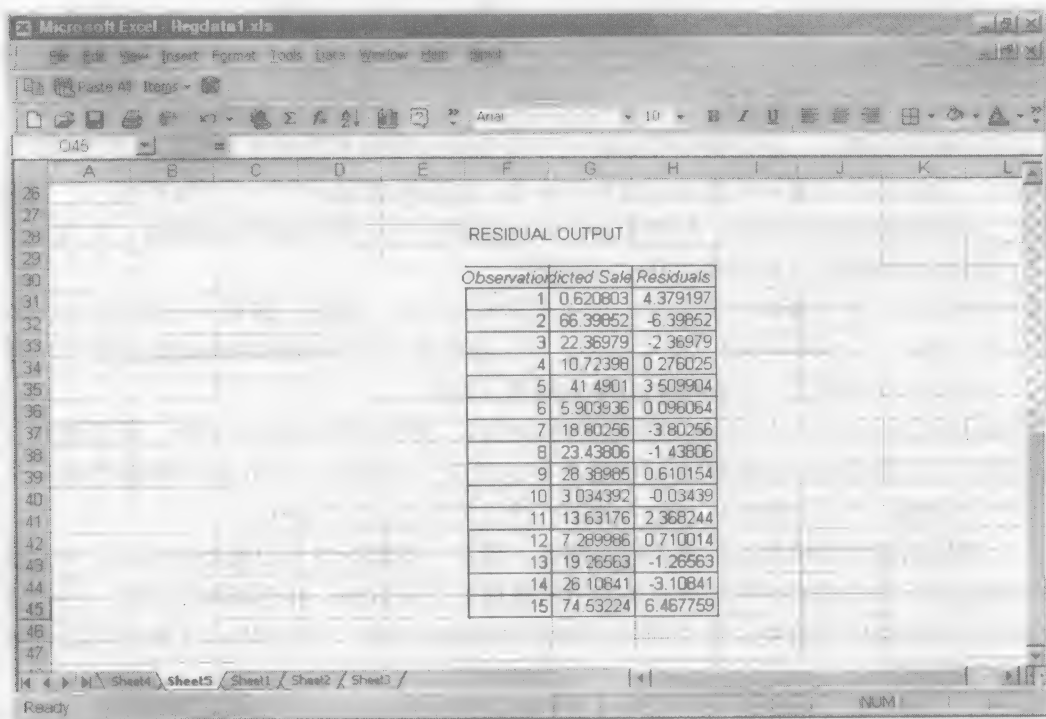
$$b_6 = 0.06594$$

将这些值代入公式 9.3 中，我们可以将其写为(所有系数四舍五入至 2 位小数)：

$$\text{销售} = -3.17 + 0.23(\text{市场潜力}) + 0.82(\text{经销商}) + 1.09(\text{销售人员}) - 1.89 \\ (\text{竞争者活动}) - 0.55(\text{服务人员}) + 0.07(\text{已有客户})$$

图 9-5 中的 P-Value 列基于显著性水平为 0.1(置信度水平 = 90%) 提供了自变量对输出结果的显著性度量。在我们的例子中，“市场潜力”和“销售人员”两个变量的 P-Value 值低于 0.10。因此，当显著性水平为 0.1 时，只有这些变量对销售有显著的影响。

使用 Excel，我们可以对 15 个区域的下一个周期的销售进行预测。为此，在回归对话框(参考图 9-4)里选中 Residual 复选框。我们可以得到图 9-6 所示的预测结果。



Observation	Predicted Sale	Residuals
1	0.620803	4.379197
2	66.39852	-6.39852
3	22.36979	-2.36979
4	10.72398	0.276025
5	41.4901	3.509904
6	5.903936	0.096064
7	18.80256	-3.80256
8	23.43806	-1.43806
9	28.38985	0.610154
10	3.034392	-0.03439
11	13.63176	2.368244
12	7.289986	0.710014
13	19.26563	-1.26563
14	26.10841	-3.10841
15	74.53224	6.467759

图 9-6 残差分析

如果小心地避免其局限性，简单或多元回归分析是一种用于解释和预测的有用方法。一般而言，不应使用这种方法来预测和构建模型时所使用的的数据值域相差甚远的值。而且，应尽可能地避免模型中的自变量之间有较高的相关性。

9.4 逻辑斯谛回归

逻辑斯谛(Logistic)回归拓展了多元线性回归的思想,处理因变量 y 是二值的情形(为简单起见,我们通常用0和1对这些值进行编码)。和多元线性回归一样,自变量 x_1, x_2, \dots, x_k 可以是分类变量、连续变量或二者的混合类型。我们用一些例子来说明[1]。

例子1:市场调研。表9-6中的数据来自AT&T公司在美国所做的调查,是其注册用户家庭中的一个全国范围内的样本。调查的目的是找出接受一种新的电信服务与教育、居住稳定性和收入的关系。

表 9-6 逻辑斯谛回归分析的数据格式

x_1	x_2	x_3	样本数量	接受者数量	不接受者数量	接受者比率
0	0	0	2160	153	2007	0.071
0	0	1	1363	147	1216	0.108
0	1	0	1137	226	911	0.199
0	1	1	547	139	408	0.254
1	0	0	886	61	825	0.069
1	1	0	1091	233	858	0.214
1	0	1	1925	287	1638	0.149
1	1	1	1415	382	1033	0.27
			10524	1628	8896	1

注意,样本数据中(新服务)的总体接受新服务的概率是 $1628/10524=0.155$ 。但是,接受新服务的概率因教育、居住稳定性和收入等自变量的类别不同而异。最低值是0.069,来自低收入、无迁居并且受过某种高等教育的家庭。而最高值是0.270,来自高收入、有迁居并且受过某种高等教育的家庭。标准多元线性回归模型不适合对这种数据建模,原因如下:

1)模型的预测概率可能会超出0~1的范围。

2)因变量并非正态分布。事实上,二项式模型更合适。例如,如果一个单元总共11户,则该变量只能取11个不同值0,1,2,...,11。想象单元中的家庭的响应通过随机掷硬币确定,正面朝上代表接受,正面朝上的概率随单元变化。

3)如果我们认为正态分布是二项式模型的近似,那么在所有的单元中,因变量的方差不是常数:对于接受新服务的概率 p 接近0.5的单元,方差比 p 在0或1附近的那些单元高。该方差还随一个单元中的住户数 n 增加。该方差等于 $n[(1-p)]$ 。

引入逻辑斯谛回归模型可以解决这些问题。现在,逻辑斯谛回归模型已经成为计量经济学中描绘选择行为以及流行病学中对风险因素建模的常用方法。在选择行为方面,已经证实逻辑斯谛回归模型作为消费者行为的标准经济学理论的扩展,符合Manski提出的随机效用理论(random utility theory)。

本质上,消费者理论是说当一个消费者面临一组选择的时候,其所做的选择具有最高的效用(效用是以任意的零点和尺度对价值的量化度量)。它假定消费者对选择列表有一个倾向性排序,而这一排序满足一些合理的标准,如传递性。倾向性排序可以基于个体(如例子1中的社会经济特征)或所做选择的属性。随机效用模型认为一个选择的效用是纳入一个随机元(random element)。当我们将随机元建模,认为其来自一个“合理的”分布时,我们可以逻辑地导出预测选择行为的逻辑斯谛模型。

如果我们以 $Y=1$ 表示选择一个选项,而 $Y=0$ 表示不选择该项,则逻辑斯谛回归模型定

义为:

$$\text{Probability}(Y=1 | x_1, x_2, \dots, x_k) = \frac{\exp(\beta_0 + \beta_1 * x_1 + \dots + \beta_k * x_k)}{1 + \exp(\beta_0 + \beta_1 * x_1 + \dots + \beta_k * x_k)}$$

其中 $\beta_0, \beta_1, \beta_2, \dots, \beta_k$ 是类似于多元线性回归模型的未知的常数。

我们模型中的因变量是:

$x_1 \equiv$ (教育: 高中或以下 = 0, 大专或以上 = 1)

$x_2 \equiv$ 居住稳定性: 近五年没有变化 = 0, 近五年有变化 = 1)

$x_3 \equiv$ 收入: 低 = 0, 高 = 1)

表 9-6 中的数据以回归程序所要求的典型格式组织。

这个例子的逻辑斯谛模型是:

$$\text{Probability}(Y=1 | x_1, x_2, x_3) = \frac{\exp(\beta_0 + \beta_1 * x_1 + \beta_2 * x_2 + \beta_3 * x_3)}{1 + \exp(\beta_0 + \beta_1 * x_1 + \beta_2 * x_2 + \beta_3 * x_3)}$$

通过下面的式子我们得到系数的有用解释:

$$\begin{aligned} \exp(\beta_0) &= \frac{\text{Prob}(Y=1/x_1=x_2=x_3=0)}{\text{Prob}(Y=0/x_1=x_2=x_3=0)} \\ &= \text{基本情况}(x_1=0, x_2=0, x_3=0) \text{接受新服务的几率} \\ \exp(\beta_1) &= \frac{x_1=1, x_2=x_3=0 \text{ 时接受新服务的几率}}{\text{在基本情况下接受服务的几率}} \\ \exp(\beta_2) &= \frac{x_2=1, x_1=x_3=0 \text{ 时接受新服务的几率}}{\text{在基本情况下接受服务的几率}} \\ \exp(\beta_3) &= \frac{x_3=1, x_1=x_2=0 \text{ 时接受新服务的几率}}{\text{在基本情况下接受服务的几率}} \end{aligned}$$

逻辑斯谛模型是下面形式的几率乘积:

对于给定的 x_1, x_2, x_3 , 接受新服务的几率 = $\exp(\beta_0) \times \exp(\beta_1 x_1) \times \exp(\beta_2 x_2) \times \exp(\beta_3 x_3)$

= {基本情况的几率} \times {关于 x_1 的因子} \times {关于 x_2 的因子} \times {关于 x_3 的因子}

如果 $x_1=1$, 那么无论 x_2 和 x_3 的取值如何, 接受新服务的几率都将被乘以相同的因子。同样, 关于 x_2 和 x_3 的因子也不随其他变量的取值而改变。变量的这个因子说明了该变量的存在对接受新服务的几率的影响。

如果 $\beta_i=0$, 那么相应(变量)的因子没有作用(乘以 1)。如果 $\beta_i < 0$, 则(变量的)因子降低了接受新服务的几率(以及概率), 而当 $\beta_i > 0$ 时, (变量的)因子增加了接受新服务的概率。

这些(值的)最大似然估计需要一个计算机程序迭代地计算。一个典型的计算机程序的输出结果见表 9-7。

表 9-7 逻辑斯谛回归分析的输出

变量	系数	标准差	p-值	为几率设置 95% 的置信区间		
				几率	下界	上界
常数 β_0	-2.5	0.058	0	0.082	0.071	0.095
x_1	0.161	0.058	0	1.175	1.048	1.316
x_2	0.992	0.056	0	2.698	2.416	3.013
x_3	0.444	0.058	0	1.56	1.393	1.746

根据系数的估计值,对于具有自变量值 x_1, x_2, x_3 的家庭,接受新服务的估计概率为:

$$\text{Probability}(Y=1 | x_1, x_2, x_3) = \frac{\exp(-2.500 + 0.161 * x_1 + 0.992 * x_2 + 0.444 * x_3)}{1 + \exp(-2.500 + 0.161 * x_1 + 0.992 * x_2 + 0.444 * x_3)}$$

用该模型估计采用新服务的家庭个数是具有自变量值 x_1, x_2, x_3 的家庭总数乘以上面的概率。表 9-8 给出了自变量取值的各种组合所对应的接受新服务者的估计值。

表 9-8 基于逻辑斯谛回归模型的估计输出

x_1	x_2	x_3	样本数	接受者数量	估计的接受者数量	接受者比例	估计概率 ($Y=1 x_1, x_2, x_3$)
0	0	0	2160	153	164	0.071	0.076
0	0	1	1363	147	155	0.108	0.113
0	1	0	1137	226	206	0.199	0.181
0	1	1	547	139	140	0.254	0.257
1	0	0	886	61	78	0.069	0.088
1	1	0	1091	233	225	0.214	0.206
1	0	1	1925	287	252	0.149	0.131
1	1	1	1415	382	408	0.27	0.289

在数据挖掘应用中,我们有在拟合模型时没有使用的保留数据作为验证数据。不妨假设我们具有以下(见表 9-9)由 598 个用户组成的验证数据。

表 9-9 检验数据

x_1	x_2	x_3	验证样本数	验证样本中 接受者数量	估计的接受者 数量	误差 (实际估计)	绝对值误差
0	0	0	29	3	2.2	-0.8	0.8
0	0	1	23	7	2.61	-4.39	4.39
0	1	0	112	25	20.302	-4.698	4.698
0	1	1	143	27	36.705	9.705	9.705
1	0	0	27	2	2.374	0.374	0.374
1	1	0	54	12	11.145	-0.855	0.855
1	0	1	125	13	16.338	3.338	3.338
1	1	1	85	30	24.528	-5.472	5.472
总计			598	119	116.202		

总体误差是 -2.8 个(新服务的)接受者,或者说估计接受者的误差百分比是 $-2.8/119 = 2.3\%$ 。平均绝对值误差百分比是: $(0.800 + 4.390 + 4.698 + 9.705 + 0.374 + 0.855 + 3.338 + 5.472)/119 = 0.249 = 24.9\%$ 。表 9-10 列出了验证数据集中的用户家庭的混淆矩阵。

表 9-10 混淆矩阵

预测	观测到的接受者	非接受者	总数
接受者	103	13	116
非接受者	16	466	482
总数	119	479	598

和多元线性回归一样,我们可以通过引入由相互作用的因子计算而得到的(新的)因子来构建更复杂的模型来反映自变量之间的相互影响。例如,如果我们认为 x_1 和 x_2 之间存在相互影响的效果,我们可以增加一个相互影响项 $x_4 = x_1 \times x_2$ 。

9.5 k-最近邻分类

k-最近邻算法构建分类模型时不对联系因变量(响应变量) y 和自变量(预测变量) $x_1, x_2, x_3, \dots, x_p$ 的函数形式 $y=f(x_1, x_2, x_3, \dots, x_p)$ 做任何假设。我们所做的唯一假定是这个函数是“光滑”函数。k-最近邻算法是非参数化(non-parametric)方法,因为与我们在线性回归中所做的不同,它不涉及假定的函数的参数估计。训练数据的每一个观测有一个 y 值,说明该观测的所属类别。例如,如果有两个类,则 y 是一个二元变量。k-最近邻方法是在训练数据集中动态地确定 k 个与我们希望分类的新观测($u_1, u_2, u_3, \dots, u_p$)相似的观测,并使用这些观测把新观测分到某一类 C 中。如果我们知道函数 f ,那么只计算 $C=f(u_1, u_2, u_3, \dots, u_p)$ 。如果我们只假设 f 是一个光滑函数,那么一个合理的想法就是在训练数据中寻找和它(根据自变量)接近的观测,然后用这些观测对应的 y 值计算 C 。这类似于插值的思想,如同我们使用正态分布表常做的那样。当我们谈到近邻时,通常意味着能够根据自变量计算观测点间的距离或相异性度量。目前,我们使用最常见的距离度量:欧几里德距离。

点 $(x_1, x_2, x_3, \dots, x_p)$ 和 $(u_1, u_2, u_3, \dots, u_p)$ 之间的欧几里德距离为 $\sqrt{(x_1 - u_1)^2 + (x_2 - u_2)^2 + \dots + (x_p - u_p)^2}$ 。当讨论聚类方法的时候,我们会考虑在预测变量空间中定义点的距离的其他方法。最简单的是当 $k=1$ 的情况,这时我们找出最近的观测点(最近邻),并且 $C=y$,其中 y 是最近邻的类别。一个值得注意的事实是:当我们的训练集中观测点的数目很大时,使用最近邻对观测分类这种简单、直观的想法可能是非常有效的。可以证明1-NN的误分类概率不劣于我们知道每个类的精确的概率密度函数时误分概率的2倍。换句话说,如果有大量的数据和充分复杂的分类规则,我们最多能将分类错误减少到使用简单的1-NN规则时的一半。下面我们将1-NN的想法拓展到k-NN。

首先,寻找 k 个最近邻,然后用多数表决规则对新的观测分类。 k 值较高的优点是能提供平滑的分类,降低由于训练数据中存在噪声而过分拟合的风险。在典型的应用中, k 是一位数或两位数,而不是成百上千。注意,如果 $k=n$,即等于训练数据集中的观测数目,则我们只不过是对所有的观测都用训练数据中的多数类来预测,而不管 $(u_1, u_2, u_3, \dots, u_p)$ 的值是什么。这显然是一个过平滑的例子,除非自变量中根本就没有关于因变量的信息。

例子 一个乘坐式割草机的制造商希望找到一种方法把一个城市中的家庭分类为可能买乘坐式割草机的家庭和不想买乘坐式割草机的家庭。在这个城市中,随机抽取12个拥有乘坐式割草机的家庭和12个没有乘坐式割草机的家庭。这些数据见表9-11和图9-7。

表 9-11 乘坐式割草机数据

观测	收入(千美元)	草地面积(千平方英尺)	拥有=1 不拥有=2
1	60	18.4	1
2	85.5	16.8	1
3	64.5	21.6	1
4	61.5	20.8	1
5	87	23.6	1
6	32	19.2	1
7	108	17.6	1
8	82.8	22.4	1
9	69	20	1
10	93	20.8	1

(续)

观测	收入(千美元)	草地面积(千平方英尺)	拥有=1 不拥有=2
11	51	22	1
12	81	20	1
13	75	19.6	2
14	52.8	20.8	2
15	64.8	17.2	2
16	43.2	20.4	2
17	84	17.6	2
18	49.2	17.6	2
19	59.4	16	2
20	66	18.4	2
21	47.4	16.4	2
22	43	18.8	2
23	51	14	2
24	63	14.8	2

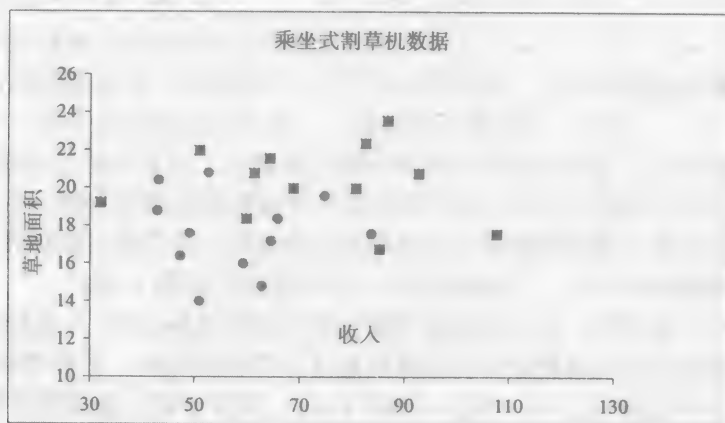


图9-7 乘坐式割草机数据图(检验和训练数据)

我们如何来选择 k 值呢? 在数据挖掘中, 对不同的 k 值, 我们用训练数据对案例(case)分类, 并计算相应的分类错误率。在这个例子中, 我们随机地把数据集划分为含有 18 个实例的训练集和含有 6 个实例的验证集。当然, 在实际的数据挖掘情况下, 会有更大规模的数据集。验证集包含表 9-11 中第 6、7、12、14、19、20 个实例。剩下的 18 个观测构成训练数据。图 9-7 展示了在训练集和验证集中的所有实例。注意, 如果我们选择 $k=1$, 那么就选择了一种对数据的局部特征非常敏感的分类方式。另一方面, 如果我们选择大的 k 值, 则相当于对大量数据点取平均, 同时平滑掉因单个数据点的噪声而导致的波动性。如果选择 $k=18$, 在各种情况下我们将只预测在数据集中最频繁出现的类。这是非常稳定的预测, 但它完全忽略了自变量中的信息。

表 9-12 列出了针对不同的 k 值在验证数据集中对观测的误分类率。

表 9-12 错误率

	k	1	3	5	7	9	11	13	18
误分类率	%	33	33	33	33	33	17	17	50

在这个例子中，我们将选择 $k=11$ (或许是 13)。这个选择很好地对在低 k 值时的变动性和高 k 值时的过平滑现象进行了折衷。值得一提的是：通过“有效参数的数目”概念推定 k 值是有益的。 k 的相应有效参数的数目为 n/k ，其中 n 是训练数据集中观测的数目。因此， $k=11$ 的有效参数数目大约为 2，光滑程度上和两参数的线性回归相似。

9.5.1 k-近邻预测

可以很容易地推广 k -NN 分类的思想，用来预测连续值(和我们建立多元线性回归模型的目的一样)，这可通过简单地预测 k 个近邻的依赖变量的平均值来实现。通常使用加权平均，权重随着与待预测的点的距离增加而减小。

9.5.2 k-NN 算法的缺点

在实际应用 k -NN 方法时有两个困难。首先，虽然从训练数据中估计参数不需要时间，但在大训练集寻找最近邻的时间可能非常长。已经实现了许多的想法去克服这个困难。主要的想法有：

- 1) 使用降维技术(如主成分分析)来减少维数，从而减少计算距离所用的时间。
- 2) 用复杂的数据结构(如搜索树)来加快最近邻的确定速度。这个方法经常通过设定“几乎最近邻”的目标来提高搜索速度。
- 3) 编辑训练数据，删除训练集中的冗余和“几乎冗余”的点，从而加快最近邻的搜索速度。例如，训练数据集中被属于同类的观测点包围的观测点对分类没有影响，则可以删除这些观察点。

其次，训练数据集所需的观测的数目随着维数 p 的增长以指数方式增长。这是因为除非训练数据集的大小随着 p 以指数方式增长，否则到最近邻的期望距离随着 p 急剧上升。这种现象被称为“维灾难”，如果在训练数据中的自变量均匀地分布在 p 维单位超立方体中，那么一个点落在距中心的 0.5 单位的概率是：

$$\frac{\pi^{p/2}}{2^{p-1} p \Gamma(p/2)}$$

表 9-13 用来说明对于不同的 p 和 n 的组合，训练集的规模如何快速下降到接近零。

表 9-13 作为 p, n 的函数，距离在 0.5 之内的点数

N	P							
	2	3	4	5	10	20	30	40
10000	7854	5236	3084	1645	25	0.0002	2×10^{-10}	3×10^{-17}
100000	78540	52360	30843	16499	249	0.0025	2×10^{-9}	3×10^{-16}
1000000	785398	523600	308452	164993	24900	0.0246	2×10^{-8}	3×10^{-17}
10000000	7853982	523600	3084251	164934	24904	0.2461	2×10^{-7}	3×10^{-14}

对于所有分类、预测和聚类方法而言，维灾难都是一个主要问题[5, 9]。这就是为什么我们经常通过诸如为模型选择预测变量的子集或采用主成分分析、奇异值分解和因子分析等方法来组合它们，努力寻找减少预测变量空间维数的方法的原因。在人工智能和数据挖掘文献中，降维通常是指因子选择(factor selection)。

9.6 GMDH

9.6.1 引言

传统的决策树方法,如第4章介绍的ID3[3,4]、CHAID和CART中的分类树都是基于数据空间的轴平行划分。大多数情况下,这类决策树的复杂度非常高。最近25年,研究者们发表了很多论文来讨论高级的决策树归纳方法。这些研究的主要目的是提高决策树预测准确度并降低构建决策树复杂度。

建立更好的决策树的一种方法是基于新属性构建决策树[7]。构建新属性提供了更快、更好的数据分类。OC1[6]方法使用基本属性的线性或多项式组合来构建新属性。OC1被称为斜决策树(oblique decision tree)方法。与传统的正交(轴平行)决策树相比,OC1方法分类的准确度更高。对于二叉决策树而言,以基本属性的并(conjunction)、交(disjunction)或反(negation)来构建新属性的方法是很有用的。以Fringe方法[8]为例,它利用节点的约束,直接从决策树分支构建新属性。

分类问题的图形解释非常简单。当属性空间被划分为区域,而数据集中的每一条记录都根据其数据记录的类别被放置到其对应的区域后,分类任务就完成了。所有改变基本属性或增加新属性的决策树方法都是通过变换基本属性空间来减少分类区域的数目。

GMDH(Group Method of Data Handling, 数据处理群组方法)的主要目的是变换基本属性空间使决策树复杂度降到最低,并提高预测准确率。在这一节中,我们将探讨如何确定构建新属性所需的基本属性的个数,使得新属性可以正确地根据已分类数据记录的类别将基本属性空间划分到相应的区域。很明显,属性的线性组合方法太简单,不能完成这一任务。因此,本节的任务是探讨使用属性的多项式组合来构建新属性的可能性。

传统的Quinlain决策树归纳方法ID3在决策树的任意一个节点上使用一个属性。对这样的属性的选择通常基于该属性的信息增益。传统决策树归纳方法的主要缺点是忽略了属性间的相关性,因为任何属性的分析都独立于其他属性。一种考虑了属性间依赖性的著名方法是非线性决策树(Non-Linear Decision Tree, NLDT)。这种方法使用属性的非线性组合来构建决策树节点的约束。

非线性决策树方法的缺点之一是在整个属性集上只使用一个预定义的非线性组合,通常是属性的多项式组合。

该方法的另一个缺点是所使用的属性组合函数的阶不高于2,因为更高阶的函数所需的计算量会更大。第三个缺点是需要构建决策树然后再抽取决策规则。

9.6.2 数据处理群组方法的背景

使用数据处理群组方法(GMDH)是为了确定属性间的相关性。这一方法是由A. G. Ivakhnenko[10]在30年前提出的。GMDH方法的应用非常广,包括多维过程逼近、单步和多步预测、数据聚类、模式和对象识别以及诊断问题求解。

GMDH方法的主要目的是识别对象的完整模型,使其满足外部的约束。

$$\varphi = f(x_1, x_2, \dots, x_n) \quad (9.4)$$

GMDH方法选择描述属性之间联系的最佳属性和最佳函数。

该方法的核心思想是将完整目标模型的搜寻分为许多阶段。在每一个阶段计算所有可能的任意属性对的函数,并且估计它们的拟合度。

因此,在第一阶段计算下面的函数:

$$y_1 = f(x_1, x_2), y_2 = f(x_1, x_3), \dots, y_s = f(x_{s-1}, x_n), \quad (9.5)$$

其中 $s = C_n^2$ 是这种函数的个数。

在该方法的第二阶段,对下列函数进行搜索:

$$z_1 = f(y_1, y_2), z_2 = f(y_1, y_3), \dots, z_p = f(y_{s-1}, y_s) \quad (9.6)$$

其中 $p = C_p^2$ 是这种函数的个数。

所有的 GMDH 方法只是函数 f 的形式不同。通常使用的是二次多项式函数。对属性个数为 n 的目标而言,一般可以通过下面的多项式来近似:

$$f(x_1, x_2, \dots, x_n) = \alpha_0 + \sum_{i=1}^n \alpha_i x_i + \sum_{i=1}^n \sum_{j=1}^n \beta_{ij} x_i x_j + \sum_{i=1}^n \sum_{j=1}^n \sum_{k=1}^n \gamma_{ijk} x_i x_j x_k + \dots \quad (9.7)$$

使用上面的多项式函数在某些情形下可能会引起一些问题。首先,必须选择多项式[6]中的哪些项是重要的,哪些项是不重要的。其次,如果对象的属性很多(例如有 10 个),则多项式的项数会非常多,因此,函数会非常复杂而且难以计算。

GMDH 方法的主要优点是使用式(9.7)来确定对象的完整模型可以用使用式(9.5)和式(9.6)确定的对象完整模型迭代合成来替代。

每一个函数都是基本分类器(elemental classificatory),其系数通过使用训练数据集和最小二乘法来确定。然后,根据外部约束从属性对的整个函数集上选择最佳的函数。每一个被选中的分类器将作为近似过程下一个阶段的参数。递归过程持续到满足总体的训练约束为止。

使用 GMDH 迭代方法寻找近似函数的过程如图 9-8 所示。

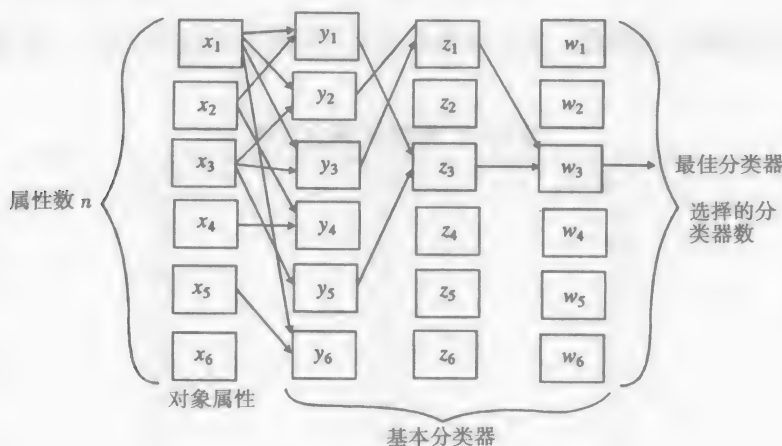


图 9-8 GMDH 方法网络

分类使用的函数类型如下:

$$F(x_1, x_2) = a_0 + a_1 x_1 + a_2 x_2 + a_3 x_1 x_2 \text{ (协变线性的)} \quad (9.8)$$

$$F(x_1, x_2) = a_0 + a_1 x_1 + a_2 x_2 + a_3 x_1 x_2 + a_4 x_1^2 + a_5 x_2^2 \text{ (二次的)} \quad (9.9)$$

$$F(x_1, x_2) = \frac{a_0 + a_1 x_1 + a_2 x_2 + a_3 x_1 x_2 + a_4 x_1^2 + a_5 x_2^2}{1 + b_1 x_1 + b_2 x_2 + b_3 x_1 x_2 + b_4 x_1^2 + b_5 x_2^2} \text{ (多项式的)} \quad (9.10)$$

GMDH 方法和人工神经网络非常类似。在这两种方法中,被分析的对象都被作为“黑箱”,而且最终的近似函数都通过许多基本函数的组合来得到。

正如图 9-8 所示,在(寻找近似函数)过程的第一个阶段,为下一个阶段选择了一些最好的函数。如果对象有 6 个属性,那么可能的函数组合个数是 15 个。从所有函数中选择出预定义个数的函数用于下一阶段。

最佳函数的选择给出了关于最佳属性对的信息,这一信息描述了对象输出属性的最佳相关性。因此,在每个下一阶段的近似过程中仅使用那些更好描述对象本性的属性对。

9.6.3 构建决策规则

决策树归纳方法是一种分类方法,因此研究的主要目标是调整 GMDH 方法使之适用于分类。一个很重要的工作是找出根据数据集产生分类规则、而不构建决策树的方法。其基本思想是用 GMDH 方法为数据域中的每一类构建近似函数。这一方法可通过以下步骤实现:

- 数据域是一张有 n 列的表,表的第 n 列包含类变量的值。整个表按类值排序,然后表被分为许多较小的表。每个较小的表只包含一个类的数据。换句话说,对数据集中的每一个类,准备一个单独的表包含相应的数据记录。这种表的最小个数等于一个类的两两可能值。
- 对数据集的每一个表,使用 GMDH 方法来寻找近似函数。同时,为每个类值识别出最佳的属性对。对每个类,这样的属性对可能是不同的。
- 如果为每个类值都确定了近似函数,那么可以在不构造决策树的情况下构建分类规则。

例子:我们使用气象数据,其中分类任务是“打或不打高尔夫球”。数据集如表 9-14 所示。

表 9-14 数据域“高尔夫球”

天气	气温	湿度	有风	类别
多云	83	78	无	打高尔夫球
多云	64	65	有	打高尔夫球
多云	81	75	有	打高尔夫球
多云	72	90	无	打高尔夫球
雨	70	96	无	打高尔夫球
雨	68	80	无	打高尔夫球
雨	75	80	无	打高尔夫球
雨	65	70	有	不打高尔夫球
雨	71	80	有	不打高尔夫球
晴	69	70	无	打高尔夫球
晴	75	70	有	打高尔夫球
晴	85	85	无	不打高尔夫球
晴	80	90	有	不打高尔夫球
晴	72	95	无	不打高尔夫球

GMDH 方法只适用于属性的连续值,因此必须改变数据集,以连续值替代所有属性和类别的符号值。经过编码的数据集如表 9-15 所示。

表 9-15 编码后的数据集

天气	气温	湿度	有风	类
2	83	78	-1	1
2	64	65	1	1
2	72	90	1	1
2	81	75	-1	1
3	70	96	-1	1
3	68	80	-1	1
3	75	80	-1	1
3	65	70	1	-1
3	71	80	1	-1
1	69	70	-1	1
1	75	70	1	1
1	85	85	-1	-1
1	80	90	1	-1
1	72	95	-1	-1

天气	
晴	1
多云	2
雨	3
有风	
有	1
无	2
类别	
打高尔夫球	1
不打高尔夫球	2

准备数据集的下一步是将数据记录已排序的表根据每个类划分为三个子表。然后使用 GMDH 方法为每个表寻找近似函数。

在这个例子中使用下列函数：

$$F(x_1, x_2) = a_0 + a_1x_1 + a_2x_2 + a_3x_1x_2$$

$$F(x_1, x_2) = a_0 + a_1x_1 + a_2x_2 + a_3x_1x_2 + a_4x_1^2 + a_5x_2^2$$

$$F(x_1, x_2) = a_0 + a_1x_1 + a_2x_2 + a_3x_1x_2 + a_4x_1^2 + a_5x_2^2 + a_6x_1x_2^3 + a_7x_1^3x_2$$

所有的函数都是多项式的，分别为一次、二次和三次。三次函数中没有包括多项式的所有可能的元素。

一般来说，根据 GMDH 方法，必须先定义最高阶函数。然后，将 GMDH 方法应用于不高于所定义次数的所有可能函数。近似过程从最低阶的函数开始，然后向高阶继续。在每一步中，近似函数必须增加新的项。在达到适当的近似误差或满足其他外部条件之后，近似过程停止。完成这样的递归过程需要使用专门的软件。

使用式(9.8)定义的近似函数，可以获得图 9-9 所示的网络。

图 9-9 中的灰色框标出了属性的最佳组合。所有函数使用平均误差进行比较：

$$MAE = \frac{1}{n} \sum_{i=1}^n |f(x_i, x_j) - class|$$

其中 n 是每个类值所对应的数据记录个数。

两个类的网络结构都是相同的。平均误差值是：

- “打高尔夫球”类 $(1)8 \times 10^{-7}$
- “不打高尔夫球”类 $(-1)4 \times 10^{-7}$

现在从图 9-9 可以看出，属性的最佳组合是：“天气-湿度”和“天气-有风”。

由式(9.9)给定的近似函数所构建的计算网络如图 9-10 所示。网络只包含一层，因为最佳分类(灰色框)的平均误差远小于第一个函数。

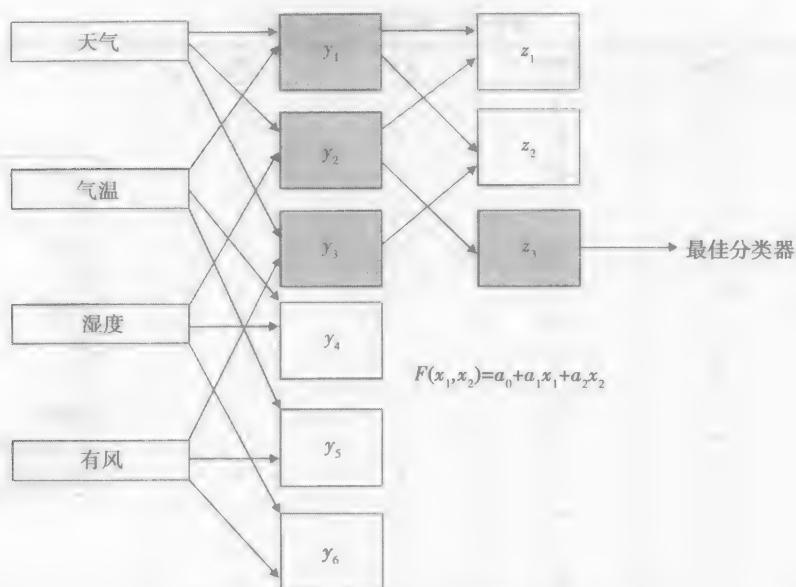


图 9-9 用式(9.8)定义的函数构建的网络

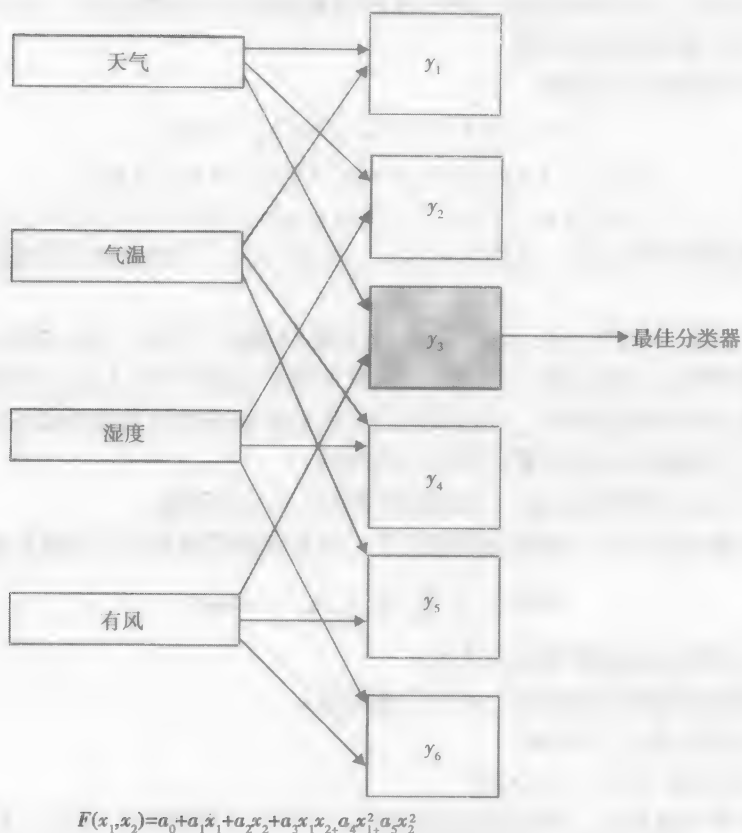


图 9-10 用式(9.9)定义的函数构建的网络

对第二个函数而言，两个类的网络结构也都是相同的。而且，第二个函数的近似平均误差是：

- “打高尔夫球”类(1) 1.9×10^{-6}
- “不打高尔夫球”类(-1) 1.5×10^{-7}

属性的最佳组合是“天气-有风”。

9.6.4 实验结果

使用 GMDH 方法进行分类任务所获得的近似函数是:

对“打高尔夫球”类(1), 平均误差(8×10^{-7})值是:

$$y_1 = f(\text{天气}, \text{气温}) = 0.99 + 1.84 \times 10^{-6} \cdot \text{天气} + 2.04 \times 10^{-7} \cdot \text{气温}$$

$$y_2 = f(\text{天气}, \text{湿度}) = 1.0 + 2.09 \times 10^{-6} \cdot \text{天气} - 6.17 \times 10^{-8} \cdot \text{湿度}$$

$$y_3 = f(\text{天气}, \text{有风}) = 1.0 + 4.22 \times 10^{-7} \cdot \text{天气} + 4.61 \times 10^{-7} \cdot \text{有风}$$

$$\text{打高尔夫球} = f(y_2, y_3) = 0.333 + 0.333y_2 + 0.333y_3$$

对“不打高尔夫球”类(-1), 平均误差(1.5×10^{-7})值是:

$$\begin{aligned} \text{不打高尔夫球} = f(\text{天气}, \text{有风}) = & 0.39 - 0.28 \cdot \text{天气} - 5.56 \times 10^{-2} \cdot \text{有风} \\ & + 5.56 \times 10^{-2} \cdot \text{天气}^2 - 0.39 \cdot \text{有风}^2 + 5.56 \times 10^{-2} \cdot \text{天气} \cdot \text{有风} \end{aligned}$$

以上获得的函数描述了数据集的每个类, 并可以用来构建分类规则。

9.6.5 讨论和总结

数据处理群组方法可以用来分析属性的相关性及其对类值的影响。

GMDH 方法还可以使用高阶多项式函数近似(属性间的)函数关系, 避免了传统近似方法中所出现的局部最小化问题。

有必要使用更大的、包含数据记录数量比近似函数阶数大很多的数据域来进行实验。本节所使用的数据域仅仅用于测试 GMDH 方法。

较小的平均误差($\sim 10^{-7}$)可能是因为数据集的记录数量少和近似函数的阶数相对高(二阶)造成的。

GMDH 方法也给出了迭代过程何时必须终止, 并且标明增加近似函数的复杂性不是不要的。在迭代终止时还可以观察到近似误差有所增加。

将来进一步的研究包括寻找近似函数估计拟合度的最佳标准。为了这一目的, 应用近似函数系数的稳定性标准是很有用的[10]。

9.7 进化计算和遗传算法

在 20 世纪 60 年代, I. Rechenberg[11]在其文章“进化策略”中引入了进化计算的思想。他的思想被其他研究者进一步发展。John Holland[12]提出了遗传算法(Genetic Algorithm, GA)并且由他本人、他的学生和同事进一步发展。这使得 Holland 于 1975 年出版了著作《自然和人工系统的自适应性》(adaption in natural and artificial systems)。

1991 年, John Koza[13]使用遗传算法来进化程序完成某些工作。它将这种方法称为遗传程序设计(Genetic Programming, GP)。在 GP 中, 使用了 LISP 程序, 因为在这种语言中程序可以以解析树(parse tree)的形式进行表达, 而解析树正是 GA 的工作对象。

最初的遗传算法是由 John Holland 于 20 世纪 70 年代在密歇根大学开发的。Holland 对生物系统可以轻松地完成印象任务印象深刻, 其轻松程度甚至超越最强大的超级计算机; 动物能够无误的识别目标, 理解和翻译声音, 并且可以几乎即时地在一个动态环境中导航。

近些年来,科学家们承诺要把生物系统的这种能力复制到机器上,但是我们开始认识到这项任务极为困难。很多科学家认为任何具有这些能力的复杂生物系统都是通过进化获得这些能力的。

9.7.1 进化理论

进化理论本身不停在进化,进化(计算)通过相对简单、自复制(self-replicating)的模块制造出具有令人惊奇能力的系统。这些模块遵循以下简单的原则:

进化发生在染色体级别:有机体本身不会进化,但是它提供了基因存在和延续的载体。染色体因基因的重组而动态变化。

本质上倾向于更多地复制那些可以产生更高适应度有机体的染色体:如果一个有机体存活了足够长的时间,并且是健康的,那么它的基因将可能通过复制延续到新一代有机体中。这一原则通常称为“适者生存”(survival of the fittest)。请记住,“适者”是相对的;一个有机体只要相较于种群中其他有机体而言是适者,它即可“生存”。

种群必须保持多样性:似乎是自然界中频繁发生的变异保证了有机体的多样性。这些基因变异经常会促使一个对于物种生存有用甚至是关键性特征的产生。由于具有可能组合的更广泛的谱系(spectrum),一个种群也更不易于感染能够完全摧毁它们的常见疾病(如病毒)或出现和繁殖有关的其他问题。

一旦我们把进化分解为这些基本的构件,将这些技巧应用于计算世界就变得更为容易,并且真正开始向着更流畅、行为更自然的机器方向前进。

Holland 开始将进化的这些性质应用于由简单数字串表示的染色体。他首先将问题编码为二进制串(0 和 1 组成的串)来表示染色体,然后用计算机生成许多这样的“位”串来形成整个种群。而且,编制了一个可以对每个位串进行评估和定秩的适合度函数(fitness function)。那些被认为是最“适合”的串之间通过一个“交叉”(crossover)例程交换数据,生成“后代”位串。Holland 甚至将他的数字染色体进行了一个“变异”操作,以在产生的子代中注入随机(因素)来保持种群的多样性。适合度函数代替了生物世界中死亡的功能,从而决定哪些串足够好得以繁殖,以及哪些串将不再保留在内存中。

程序在内存中保持一定数目的“染色体”,并且整个串“种群”持续进化直到它们将适合度函数最大化为止。然后,结果被解码,返回到原始值来揭示出解决方案。John Holland 依然是这一领域的一个活跃的先驱者,而且数以百计的科学家和学者加入,并将他们大多数时间投入到这一相对于传统的线性规划、数学和统计技术而言更有前途的领域。Holland 的原始遗传算法非常简单,但是却非常健壮,可以为多种问题找到最佳解决方案。许多今天运行的程序只对原始遗传算法进行少量修改,便可以解决大型和非常复杂的现实世界问题。

随着(对进化计算)研究兴趣不断在学术圈扩大,主流的桌面计算机也开始具有强大的计算能力,类似微软 Windows 和 Excel 的标准也使复杂模型的设计和维护变得更容易。使用实数而不是位串来表示染色体减轻了对其进行编解码工作的困难。

遗传算法日益流行,包括研讨会、书籍、杂志文章以及随处可见的咨询顾问。遗传算法国际会议已经开始关注(遗传算法的)实际应用,这正是避开其他“人工智能”技术而走向成熟标志。许多世界 500 强的公司不断使用遗传算法来解决现实世界中的问题,其范围从经纪公司到电力企业、电话公司、连锁饭店、汽车制造商和电话网络。事实上,你很有可能已经

间接使用了遗传算法。

例子 我们来看一个生物界很简单的进化例子(在很小的规模上)。这里,“进化”是指一个种群中基因分布或频率的任何变化。当然,关于进化的一个非常有趣的事情是它使种群不断适应它们的环境。

假设我们正在考察一个老鼠种群。这些老鼠按大小分为小的和大的;按颜色分为浅色和深色。种群由图 9-11 所示的八只老鼠组成。



图 9-11 初始种群

一天,猫出现在周围并开始吃老鼠。事实证明,深色的和小老鼠更难以被猫找到。因此,不同的老鼠具有不同的几率来躲避猫足够长的时间以进行繁殖。这影响了老鼠下一代的特征。假定老老鼠在繁殖后立即死亡,则下一代老鼠看起来如图 9-12 所示。



图 9-12 一代后的老鼠

注意,大老鼠、浅色老鼠,尤其是大的浅色老鼠很难存活足够长的时间来进行繁殖。这一状况又持续到下一代(如图 9-13 所示)。

现在,老鼠种群多数由小的、深色的老鼠构成,因为这些老鼠相对于其他老鼠更适合于在这种环境中生存。类似的,因为猫能吃到的老鼠越来越少而开始变得饥饿,也许喜欢以草为食物的猫更能适应环境,于是将它们喜欢草的基因传递给它们的下一代。这就是“适者生

存”的核心概念。更准确地说，是“生存到繁殖”。从进化的角度来说，在种群中作为最健康的单身汉是没有意义的，因为必须进行繁殖以使基因影响下一代。



图 9-13 最终的种群

一个数字化的例子。想象这样一个问题，有两个变量， X 和 Y 能产生结果 Z 。如果我们为每个 X 和 Y 值计算并绘制 Z 值，我们可以看到一个“地形”解出现。如果我们试图寻找 Z 的最大值，那么函数的峰值是“好的”解，而谷值是“坏的”解。

当我们使用遗传算法来最大化函数的时候，我们从随机（如图 9-14 中的黑点）生成的几个可能的解或方案（scenario）开始，而不是从一个开始。然后我们为每一个方案计算函数的输出并将每个方案绘制成一个点。接下来，将这些方案按从好到坏。我们保留最好的一半，丢弃其他的方案。

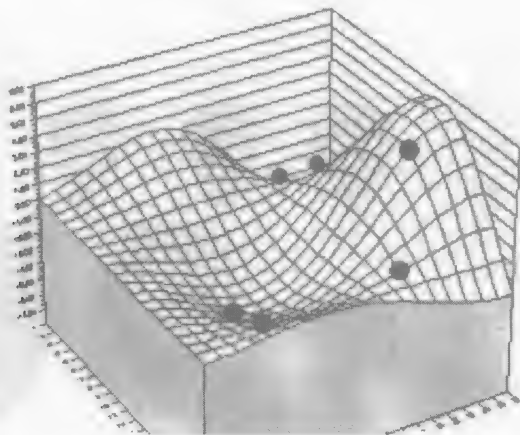


图 9-14 初始种群

保留的三个方案（选定的数据）都复制其本身，将方案的个数恢复至 6 个。下面是有趣的部分：六个方案的每一个由两个可调值（以 X 、 Y 坐标进行绘制）构成。方案之间随机地互

相配对。现在每个方案将它的第一个可调值和它的搭档的对应值进行交换。如表 9-16 所示：

这一操作称为交叉 (crossing over 或 crossover)。当我们的六个方案随机配对并进行交叉后，可以得到新一组的方案，如图 9-15 所示。

表 9-16 交叉操作

	之前	之后
方案 1	3.4, 5.0	2.6, 5.0
方案 2	2.6, 3.2	3.4, 3.2

在前面的例子中，我们假定原始的三个方案 a 、 b 和 c 和复制的 A 、 B 、 C 配对，形成 aB 、 bC 、 cA 对。这些方案对互相交换它们的第一个可调值，相当于在图 9-14 中交换两个点的 X 坐标。方案种群即完成了它的“生死”周期，生存了一代。

注意，一些新的方案的输出的位置可能比原来一代中所有的方案都低（低海拔）。但是，大多数方案都向着最高的峰值上移了，体现出了进展。如果我们让种群进化下一代，可以看到下面的情况（如图 9-15 所示）。

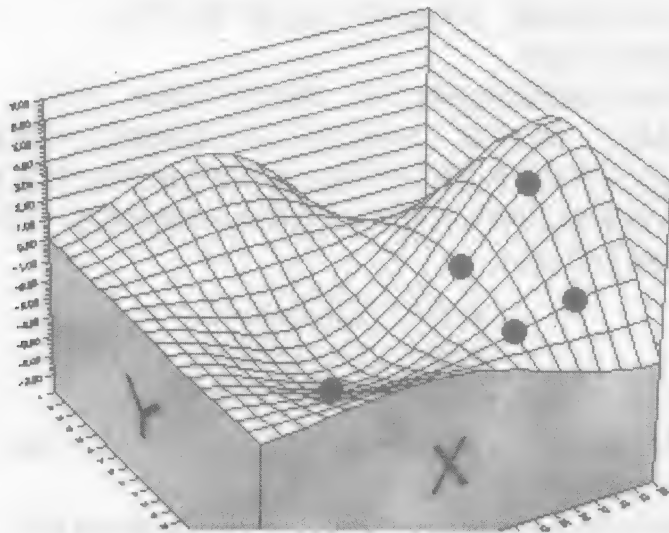


图 9-15 种群向极值逐步移动

你可以在最后一代中看到方案种群的平均性能是如何增长的。在这个例子中，（性能）可提高的空间并不是很大。这是因为每个有机体只有两个基因，只存在六个有机体，并且不能产生新的基因。这意味着基因池有限。基因池是种群中所有有机体的所有基因之和。

通过复制更多生物世界进化中的遗传力量，提高每个有机体中的基因数量，提高种群中的有机体数量，并允许偶然的、随机的变异以得到功能更强大的遗传算法。另外，不是简单地选择最好的表现者进行繁殖，还可以选择更自然的生存和繁殖方式，即随机的因素稍稍偏向于选择那些表现更好的个体（即使最大和最强壮的狮子也可能遭到雷击）。

所有这些技巧都可以刺激基因细化，还可以帮助维持基因池的多样性，并保持各类基因都是可用的以便其在不同的组合中被证明是有用的。

典型的遗传算法如图 9-16 所示。

9.7.2 遗传算法

将遗传算法应用于某个具体的问题，我们需要定义或选择下面的五个成分：

- 问题潜在解的遗传表示或编码方案。

```

procedure GA; {
    t = 0;
    initialize population P(t);
    evaluate P(t);
    until (done)
    {
        t = t + 1;
        parent_selection P(t);
        recombine P(t);
        mutate P(t);
        evaluate P(t);
        survive P(t);
    }
}

```

图 9-16 遗传算法

- 潜在解的初始种群的生成方法。
- 作为环境角色的评估函数，对解根据其“适合度”进行评价。
- 改变子代组成的遗传操作。
- 遗传算法使用的各种参数值(种群大小，操作的应用比例，等等)。

通过一个单变量例子，我们来看一下如何使用 GA 来寻找函数 $f(x)$ 的最小值。我们使用一个二元向量作为染色体来表示单变量 x 的实值。向量的长度依赖于需要的精度。

染色体编码：染色体应以某种方式包含关于其所表示的解的信息。最常用的编码方法是二进制串。染色体如图 9-17 所示。

染色体 1	1101100100110110
染色体 2	1101111000011110

图 9-17 使用二进制串对变量编码

当然，还有很多其他的编码方法。这主要和要解决的问题有关。例如，可以用整数或实数编码，有时用某种置换排列编码也很有用。

初始种群：初始化过程非常简单：我们随机地生成给定长度的染色体(二进制码)种群。

评估：对代表染色体的二元向量的评估函数等价于初始函数 $f(x)$ ，其中给定染色体表示对应实数值 x 的二进制码。

交替：在交替阶段，基于前一迭代中种群的评估选择出新的种群。个体的选择取决于其目标函数值或适合度值。在选择阶段有不同的方案可以使用。

遗传操作——交叉和变异：交叉从父母染色体中选择基因来生成新的子代。进行这一操作的最简单方式是随机选定一个交叉点，从第一个父母拷贝交叉点之前的所有基因，从第二个父母拷贝交叉点之后的所有基因。

并不需要将交叉应用于选取的所有个体对。是否进行交叉取决于一个指定的概率，称为**交叉概率(PC)**，其值通常在 0.5 ~ 1 之间。

图 9-18 显示了一个交叉过程(|是交叉点)：

染色体1	11011 00100110110
染色体2	11011 11000011110
子代 1	11011 11000011110
子代 2	11011 00100110110

图 9-18 父串的交叉产生子代

还可以用其他方法来进行交叉。例如，我们可以选择更多的交叉点。交叉可能非常复杂并且依赖于染色体的编码。对特定问题，交叉(方法)可以提升遗传算法的性能。

变异：交叉完成之后就进行变异。这是为了防止种群的所有解都落入待解问题的局部最优解。变异随机地改变新的子代。对二进制编码来说，我们可以随机选择几位从1变为0或从0变为1(参见图9-19)。

原始子代 1	1101111000011110
原始子代 2	1101100100110110
变异的子代 1	1100111000011110
变异的子代 2	1101101100110100

图9-19 通过翻转进行变异(变异的位置突出显示)

变异依赖于编码以及交叉。例如，当我们进行置换编码时，变异可能是交换了两个基因。

本书光盘中包含了一个 applet(取自互联网)。它对上述的概念进行了说明。图9-20是这个 applet 的截图。

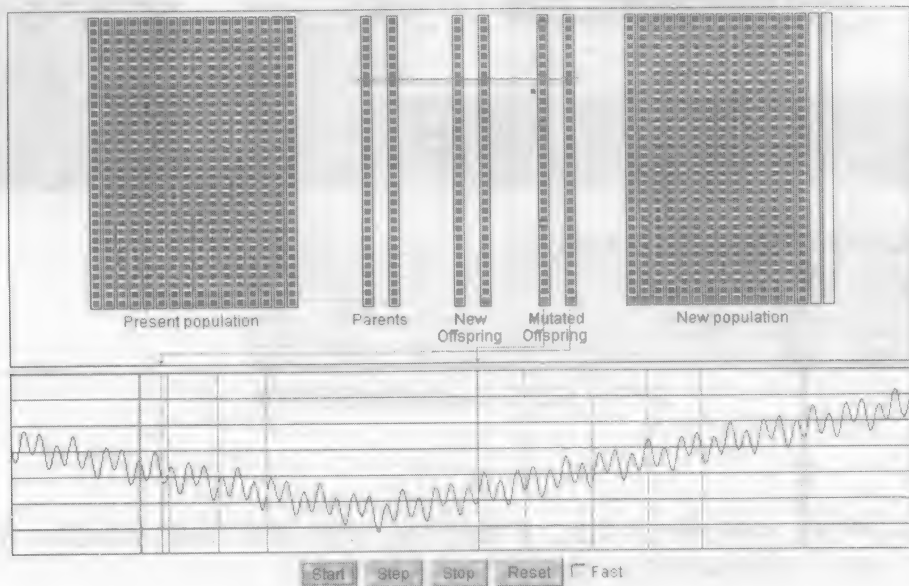


图9-20 applet 截图

9.7.3 使用遗传算法进行机器学习

优化问题是遗传算法最常见的应用领域之一。一般而言，优化问题试图通过决定生产过程中所选择特征的值来确定一个解，从而将组织的利润最大化或使生产成本最小化。遗传算法应用的另一个典型领域是发现一个给定复杂系统的输入到输出映射，这正是所有机器学习算法都试图解决的一类问题。

输入到输出映射的基本思想是生成一个合适形式的函数或模型，该函数或模型一般说来要比通常由一个输入-输出样本集表示的原始映射简单。我们认为函数最佳地描述了这一映射。对“最佳”的度量取决于具体应用。常用的度量有函数的准确性、鲁棒性和计算效率。

一般而言,确定一个满足所有条件的函数并非易事。因此,GA 可用来确定一个“好的”函数,该函数可以成功地应用于各种应用,如模式识别、控制和预测等。映射的过程可以是自动的。使用 GA 技术,这种自动化代表了归纳机器学习模型形成的另一种方法。

一个经典的例子是来自“Palisade.com”(www.palisade.com)的 Evolver 电子数据表求解器中的“交易规则发现器”(trading rule finder)。Evolver 是基于电子数据表(Microsoft Excel)求解器的 GA,可用来解决各种工程优化问题,包括数据挖掘问题。图 9-21 给出了交易规则发现器的截图。这个 Excel 电子数据表包含本书光盘中。感兴趣的读者可以下载该软件(免费一个月)并运行它。本书光盘中介绍了如何使用这个软件。

交易规则发现器基于历史数据集寻找投资的最佳规则。

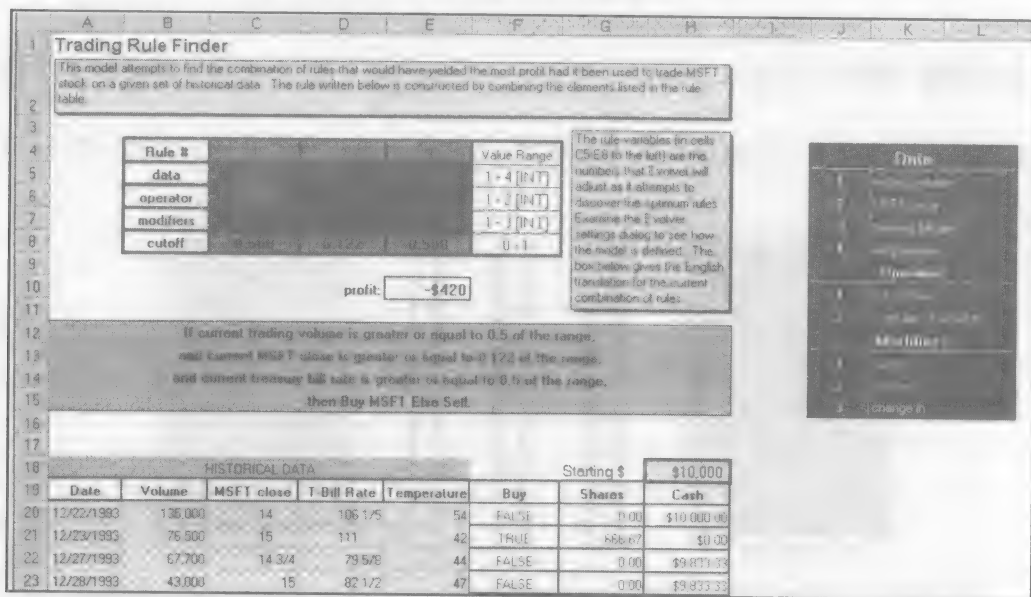


图 9-21 使用 Evolver 的交易规则发现器

习题

- 如图 9-22 所示,车辆的属性为颜色、车型和产地,变量“被窃”的值为是或否。预测属性为红色、SUV(运动型多功能车)和国产的汽车是否会被窃。

样本序号	颜色	车型	产地	被窃
1	红色	跑车	国产	是
2	红色	跑车	国产	否
3	红色	跑车	国产	是
4	黄色	跑车	国产	否
5	黄色	跑车	进口	是
6	黄色	SUV	进口	否
7	黄色	SUV	进口	是
8	黄色	SUV	国产	否
9	红色	SUV	进口	否
10	红色	跑车	进口	是

图 9-22 车辆被窃数据的训练样本

2. 假定有如图 9-23 给出的训练样本。属性 F_1 取值为 a 、 b 和 c ；属性 F_2 为布尔值；属性 F_3 的取值为 $[0, 1]$ 之间的实数值。

1) 朴素贝叶斯系统如何对如下测试样本分类？（将数值特征等分为三个区间。）

$$F_1 = c \quad F_2 = T \quad F_3 = 0.8$$

2) 讨论 2-最近邻算法如何对上题的测试样本进行分类。

3) 给出 ID3 如何在上述训练样本集上确定决策树的根节点的计算过程。

	F_1	F_2	F_3	类
样本 1	a	T	0.2	+
样本 2	b	F	0.5	+
样本 3	b	F	0.9	+
样本 4	b	T	0.6	-
样本 5	a	F	0.1	-
样本 6	a	T	0.7	-

图 9-23 习题 2 数据的训练样本(合成数据)

3. 假设希望识别一个公司产品的好坏。可以对每个产品的三个数值属性 P_1 、 P_2 和 P_3 进行度量。随机地从传送带上取若干产品进行全面测试以确定是否合格，获得如图 9-24 所示的结果。

1) 说明 3-最近邻算法如何对下述新的样本进行分类。

$$P_1 = 6.3 \quad P_2 = 5.1 \quad P_3 = 0.4$$

2) 说明如何应用朴素贝叶斯算法解决这一问题。根据贝叶斯算法，上题中的样本更可能是一个合格产品还是不合格产品？

P_1	P_2	P_3	结果
0	0.2	0.8	合格
9.2	0.7	1.5	不合格
4.9	0.1	2.9	合格
2.7	5.3	6.2	不合格
2.4	0	3.7	合格

图 9-24 习题 3 数据的训练样本(合成数据)

4. 图 9-25 所示的样本数据集包括了 12 个顾客的档案及其对新的促销邮件做出的购买或不购买反应的记录：

	顾客收入	顾客使用高速网络连接	教育水平	购买决定
1	低	否	高中	不买
2	低	是	高中	不买
3	低	否	大学	不买
4	低	是	大学	买
5	中	否	高中	不买
6	中	是	高中	不买
7	中	否	大学	买
8	中	是	大学	买
9	高	否	高中	不买
10	高	是	高中	买
11	高	否	大学	买
12	高	是	大学	买

图 9-25 习题 4 的数据集

请预测一个新的顾客的购买决定，其年收入为 15000000，使用 512KB 的调制解调器，并且在 IIM Ahmedabad 大学主修商业管理学位。

5. 表 9-17 给出了银行的一个样本数据。第二列记录了专家对每家银行财务状况的判断。后两列给出了银行财务分析中常见的比率值。

表 9-17 银行财务状况

样本	财务状况(y)	总贷款和租约/总资产(x_1)	总开销/总资产(x_2)
1	1	0.64	0.13
2	1	1.04	0.1
3	1	0.66	0.11
4	1	0.8	0.09
5	1	0.69	0.11
6	1	0.74	0.14
7	1	0.63	0.12
8	1	0.75	0.12
9	1	0.56	0.16
10	1	0.65	0.12
11	0	0.55	0.1
12	0	0.46	0.08
13	0	0.72	0.08
14	0	0.43	0.08
15	0	0.52	0.07
16	0	0.54	0.08
17	0	0.3	0.09
18	0	0.67	0.07
19	0	0.51	0.09
20	0	0.79	0.13

财务状况 = 1 财务状况差的银行

= 0 财务状况好的银行

使用 XLminer 制定逻辑斯谛回归表达式。

参考文献

- [1] Lecture Notes by Nitin R. Patel.
- [2] C. Mansky, *The Structure of Random Utility Models, Theory and Decision*, 8, pp. 229–254, 1977.
- [3] J.R. Quinlan, Induction of decision trees, In: *Machine Learning*, 1, pp. 81–106 1986.
- [4] J.R. Quinlan, *C4.5 Programs for machine learning*, San Mateo, CA, Morgan Kaufmann, 1993.
- [5] J. Mingers, An Empirical Comparison of Selection Measures for Decision-Tree Induction, In: *Machine Learning*, 3, pp. 319–342, 1989.
- [6] A. Ittner and M. Schlosser, Non-linear Decision Trees–NDT, In: *Proceedings of the 13th International Conference on Machine Learning (ICML'96)*, 1996.
- [7] J. Wnek and R.S. Michalski, Hypothesis-driven constructive induction in AQ17–HCI: A method and experiments, In: *Machine Learning*, 14, pp. 139–168, 1994.

- [8] Pagallo, Adaptive Decision Tree Algorithms for Learning from Examples, In: *Ph.D. Thesis*, University of California at Santa Cruz. CA, 1990.
- [9] Z. Zheng, Constructing New Attributes for Decision Tree Learning, In: *A Thesis Submitted in Fulfillment of the Requirements for the Degree of Doctor of Philosophy*, The University of Sydney, Australia, 1996.
- [10] A.G. Ivakhnenko and G.A. Ivakhnenko, The review of problems solvable by algorithms of the Group Method of Data Handling (GMDH), In: *Pattern Recognition and Image Analysis*, vol. 5, no. 4, pp. 527-535, 1995.
- [11] I. Rechenberg, *Evolutionstrategie: Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution*, Frommann-Holzboog, Stuttgart, 1973.
- [12] J.H. Holland, *Adaptation in Natural and Artificial Systems*, Ann Arbor, Michigan: The University of Michigan Press, 1975.
- [13] J.R. Koza, Evolving a computer program to generate random numbers using the genetic programming paradigm, *Proceedings of the Fourth International Conference on Genetic Algorithms*, 37-44, La Jolla, CA: Morgan Kaufmann, 1991.

第 10 章 支持向量机

10.1 引言

支持向量机(SVM)[1, 2]是最近出现的数据挖掘实践者使用的工具。它们是新一代的基于统计学习理论的学习系统。SVM 在两个方面非常有用。首先, SVM 学习基于优美、简单的思想, 并且对从实例中学习什么提供了非常清晰直观的解释。其次, 在文本分类、手写文字识别、图像分类、生物序列分析等实际应用中, 它具有非常好的性能。

SVM 属于监督学习算法, 在这类算法中, 为学习机提供一个样本集(或称为输入)及其相应的分类标识(或称为输出值)。与决策树一样, 样本以属性向量的形式提供, 所以输入空间是 R^n 的子集。

SVM 构建了一个分隔两类的超平面(这也可以扩展到多类问题)。在构建的过程中, SVM 算法试图使两类之间分隔达到最大化, 如图 10-1 所示。

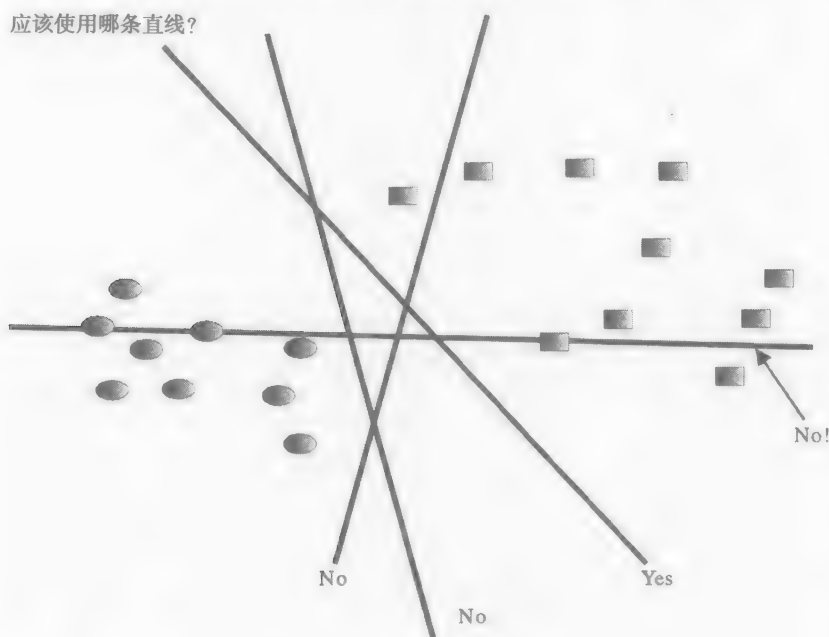


图 10-1 选择分类的最佳平面

以一个很大的边缘分隔两个类可以使期望泛化误差最小化。“最小化泛化误差”的含义是: 当对新的样本(类值未知的数据点)进行分类时, 基于学习所得的分类器(超平面), 使得我们(对其所属分类)预测错误的几率被最小化。直觉上, 这样一个分类器实现了两类之间的分离边缘最大化。图 10-2 解释了“最大化边缘”的概念。和分类器平面平行、分别穿过数据集中的一个或多个点的两个平面称为边界平面(bounding plane)。这些边界平面的距离

称为边缘(margin), 而“通过 SVM 学习”的含义是找到最大化这个边缘的超平面。

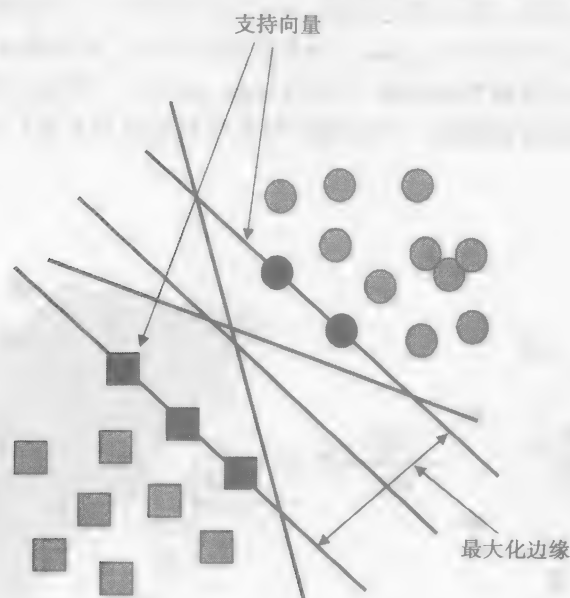


图 10-2 一个最大间隔分类器

落在边界平面上的(数据集中的)点称为支持向量(support vector)。这些点在这一理论中的作用至关重要, 故称为“支持向量机”。其中“机器”的含义是算法。

Vapnik[2, 4, 5]证明: 如果训练向量被一个最佳超平面准确无误地分隔, 那么在测试样本上的期望误差率由支持向量的个数和训练样本的个数之比来界定。由于该比值和问题的维度无关, 因此, 如果可以找到一个较小的支持向量集, 就可以保证得到很好的泛化能力。

以图 10-3 所示的数据点为例, 我们可以简单地将误分类的个数最小化, 同时针对那些被正确分类的样本来使间隔最大化(来进行分类)。也就是说, 在 SVM 训练算法中是允许有训练误差的。

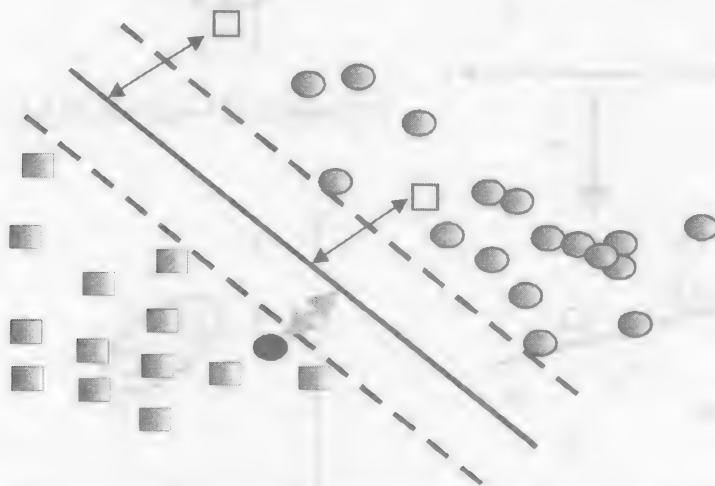


图 10-3 具有训练误差的线性分类器

如图 10-4 所示, 在有些情形下, 被聚类成两类的点并不是线性可分的。也就是说, 如果我们尝试使用线性分类器, 就必须忍受很大的“训练误差”。在这种情况下, 我们将数据非线性映射到称为特征空间 (feature space) 的更高维空间 F , 使其线性可分。为了区分这两个空间, 我们称数据点的原始空间为输入空间 (input space)。“特征空间”的超平面对应于原输入空间的一个非线性的分离曲面。因此我们称之为非线性分类器 (nonlinear classifier)。参见图 10-4。

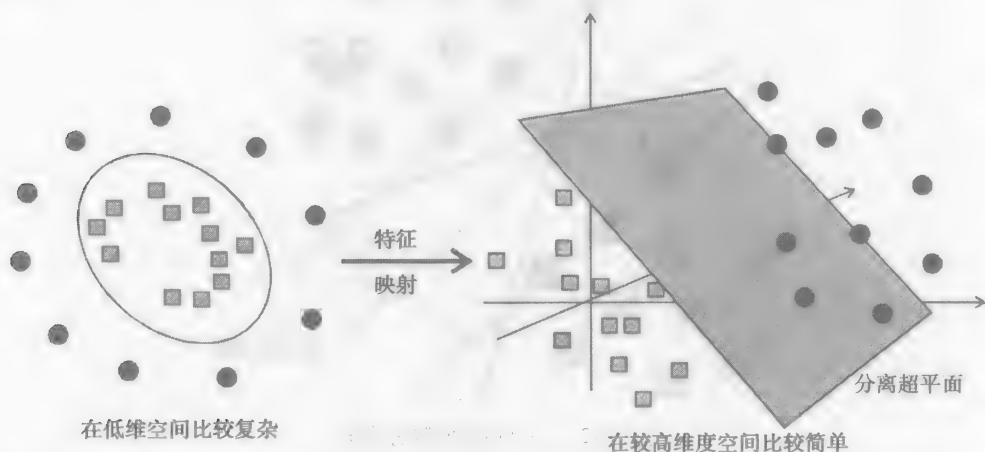


图 10-4 非线性分类器

图 10-5 进一步解释了到特征空间的非线性映射过程。注意, 二维输入点 $(x_1, x_2)^T$ 被映

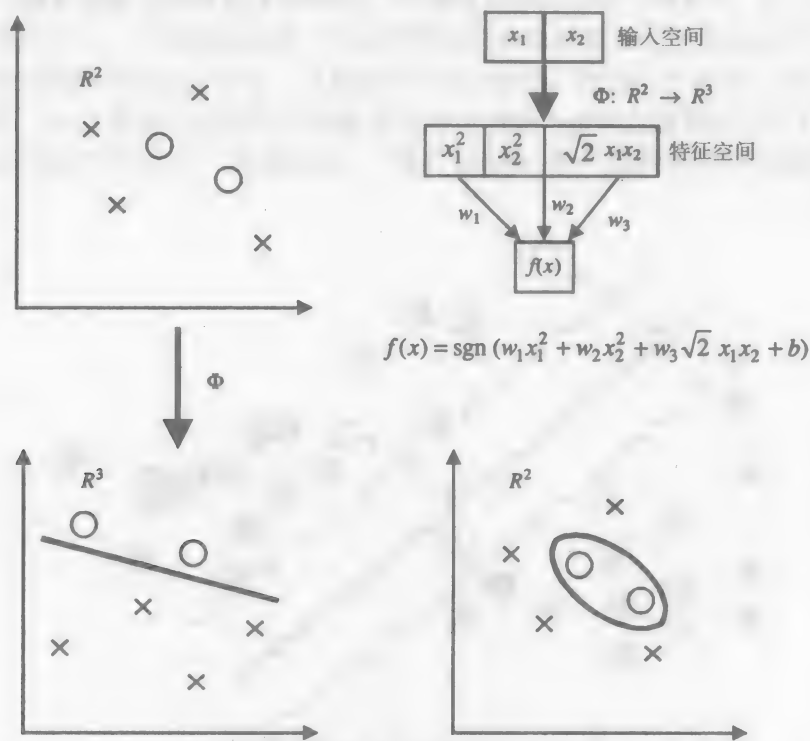


图 10-5 输入空间和特征空间

射成三维点 $(y_1, y_2, y_3)^T = (x_1^2, x_2^2, \sqrt{2}x_1x_2)^T$ 。到这里,读者可能会觉得这个过程需要很大的计算量,当数据本身是高维的,而需要将其映射到更高维(可能是无限维)的特征空间中时尤其如此。然而,稍后我们可以看到,为了得到超平面分类器,没有必要做任何显式的到高维空间的映射。所有的计算将在输入空间本身完成。

此外,存在一些可能产生不同特征空间的映射。问题是对给定的分类问题,应该采用哪个映射。Vapnik[2]回答了这一问题,必须找到使泛化误差最小化的那个映射。

文献中有许多 SVM 算法[6, 10],并且每年都会出现一些新的算法。本书无法对所有算法和其实现进行详尽的讨论。因此我们只讨论经典的 SVM 方法,而不对其实实现细节和那些容易实现的变种(简单的方法)进行介绍。Mangasarian[11, 12]的算法就是这样的变种算法,任何计算机专业的学生都可以轻松实现。这些算法甚至可用微软的 Excel 电子表格来实现。在下一节,我们使用一些例子来讨论这些算法的理论。经典 SVM 的推导过程可参见附录。

10.2 线性支持向量机的基本思想

符号约定:向量和矩阵用粗体表示。 n 维空间 R^n 中的两个向量 \mathbf{x} 和 \mathbf{y} 的内积用 $\mathbf{x}^T\mathbf{y}$ 表示。 \mathbf{x} 的范数用 $\|\mathbf{x}\|$ 表示。

如上一节所述,支持向量机分类器是基于超平面的。

$$\mathbf{w}^T\mathbf{x} - \gamma = 0 \quad \mathbf{w} \in R^n \quad (10.1)$$

其中 $\mathbf{w} = (w_1, w_2, \dots, w_n)^T$ 并且 $\gamma \in R^n$ 。

相应的决策函数是:

$$f(\mathbf{x}) = \text{sign}(\mathbf{w}^T\mathbf{x} - \gamma) \quad (10.2)$$

这里 $\mathbf{x} = (x_1, x_2, x_3, \dots, x_n)^T$ 是 n 维实数空间 R^n 中的一个向量。

在二元问题中,决策平面由下式给出:

$$w_1x_1 + w_2x_2 - \gamma = 0$$

该平面可通过求解一个约束的二次优化(constrained quadratic optimization)问题而唯一确定。其解 \mathbf{w} 可通过位于边缘(margin)上的训练模式的一个子集来展开(见图 10-2)。

$$\mathbf{w} = \sum_{i=1}^k \alpha_i \mathbf{x}_i \quad (10.3)$$

$$\mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} = \alpha_1 \begin{bmatrix} x_{11} \\ x_{12} \\ \vdots \\ x_{1n} \end{bmatrix} + \alpha_2 \begin{bmatrix} x_{21} \\ x_{22} \\ \vdots \\ x_{2n} \end{bmatrix} + \dots + \alpha_k \begin{bmatrix} x_{k1} \\ x_{k2} \\ \vdots \\ x_{kn} \end{bmatrix}$$

这一训练模式的子集称为支持向量,它包含关于分类问题的所有相关信息。省略计算的细节,我们需要强调一下这个算法的一个关键性质:二次规划问题和最终的决策函数。

$$f(\mathbf{x}) = \text{sign}(\mathbf{w}^T\mathbf{x} - \gamma) = \text{sign}\left(\sum_{i=1}^k \alpha_i (\mathbf{x}_i^T\mathbf{x}) - \gamma\right) \quad (10.4)$$

都只依赖于模式间的内积。因此我们可以严格地将问题推广到非线性的情况。

下面我们讨论图 10-6 所示的两个边界超平面 $\mathbf{w}^T\mathbf{x} - \gamma = 1$ 和 $\mathbf{w}^T\mathbf{x} - \gamma = -1$ 。

边界超平面 $\mathbf{w}^T\mathbf{x} - \gamma = 1$ 到原点的距离是 $|\gamma - 1| / \|\mathbf{w}\|$; 而 $\mathbf{w}^T\mathbf{x} - \gamma = -1$ 到原点的距离是 $|\gamma + 1| / \|\mathbf{w}\|$ 。所以,两个边界超平面的距离是 $2 / \|\mathbf{w}\|$ 。同时注意,向量是和两

个超平面垂直的。最大化距离 $2/\|w\|$ 相当于最小化其倒数, 即 $\frac{1}{2}\|w\| = \frac{1}{2}\sqrt{w^T w}$ 。

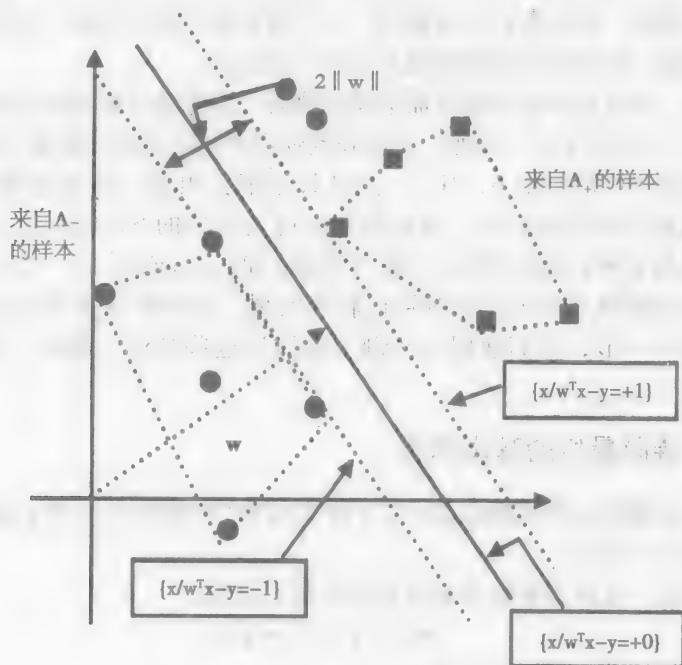


图 10-6 边界超平面

更形式地, 我们试图最小化 $\frac{1}{2}w^T w$ 。

在输入空间中, 如果数据不是线性可分的, 支持向量机通过非线性映射 $\phi: R^n \rightarrow F$ 将数据映射到某个其他点积空间(称为特征空间) F 。然后在 F 中执行上述线性算法。这只需计算点积 $\phi(x)^T \phi(x)$ (即可完成映射)。在文献中, 这一函数称为核(kernel), 我们用 $K(x, y) = \phi(x)^T \phi(x)$ 表示。

在特征空间中, 我们寻找一个进行线性分离超平面

$$w^T \phi(x) - \gamma = 0 \quad w \in F$$

使得最大化地分隔两个类。

注意, 在线性情况下, w 和 x 的维数相同。现在 w 的维数是 $\phi(x)$ 的维数。

同样, 可以证明 $w = \sum_{i=1}^k \alpha_i \phi(x_i)$, 其中 k 是支持向量个数。决策函数可以表达为:

$$f(x) := \text{sign}(w^T \phi(x) - \gamma) = \text{sign}\left(\sum_{i=1}^k \alpha_i (\phi(x_i)^T \phi(x)) - \gamma\right) \quad (10.5)$$

如果 F 的维数很高, 则计算 $\phi(x_i)^T \phi(x)$ 的开销很大。然而, 存在一些简单的核, 可进行快速计算。一个很好的例子是多项式核: $K(x, y) = (x^T y)^d$ 。

对 $d=2$ 和 $x, y \in R^2$, 有:

$$\begin{aligned} (x^T y)^2 &= [(x_1, x_2)^T (y_1, y_2)]^2 = [x_1 y_1 + x_2 y_2]^2 = 2x_1 y_1 x_2 y_2 + (x_2 y_2)^2 \\ &= [(x_1^2, x_2^2, \sqrt{2}x_1 x_2)^T (y_1^2, y_2^2, \sqrt{2}y_1 y_2)] = \phi(x)^T \phi(y) \end{aligned}$$

这里, ϕ 将 (x_1, x_2) 映射到 $[(x_1^2, x_2^2, \sqrt{2}x_1 x_2)]$ 。

这就是所谓的核技巧(kernel trick)。为了进行计算,我们实际上并没有把数据映射到特征空间。计算在输入空间本身进行。在这个例子中, $(x^T y)^2$ 等于 $\phi(x_i)^T \phi(y)$ 。 $x^T y$ 的值是标量,并且计算后,我们只要再求它的平方即可得到 $\phi(x_i)^T \phi(y)$ 的值。

求解特征空间 F 中训练样本子集(称作支持向量)的线性组合系数 w 的问题可以转换成一个求解二次规划问题。转换细节见附录。训练算法采用序列最小优化法(Sequential Minimal Optimization, SMO)。一旦使用 SMO 算法得到支持向量[9],新数据点 x 的分类只需要计算 $\left(\sum_{i=1}^k \alpha_i (\phi(x_i)^T \phi(x) - \gamma)\right)$, 其中 i 是支持向量的下标。

除了多项式核,实践中还使用径向基函数(radial basis function)核,如

$$K(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right) \quad (10.6)$$

标准 SVM 的推导可视为二次优化问题并以拉格朗日乘子(Lagrangian multipliers)求解,详见附录。

10.3 软边缘 SVM: 线性核

考虑图 10-7 所示的问题:在 n 维实数空间 R^n 中对 m 个点进行分类,以 $m \times n$ 的矩阵 A 和 $m \times m$ 的对角阵 D 表示,其中 D 的对角线上的 $+1$ 或 -1 根据对应的点 O_i 属于类 A_+ 或 A_- 确定。

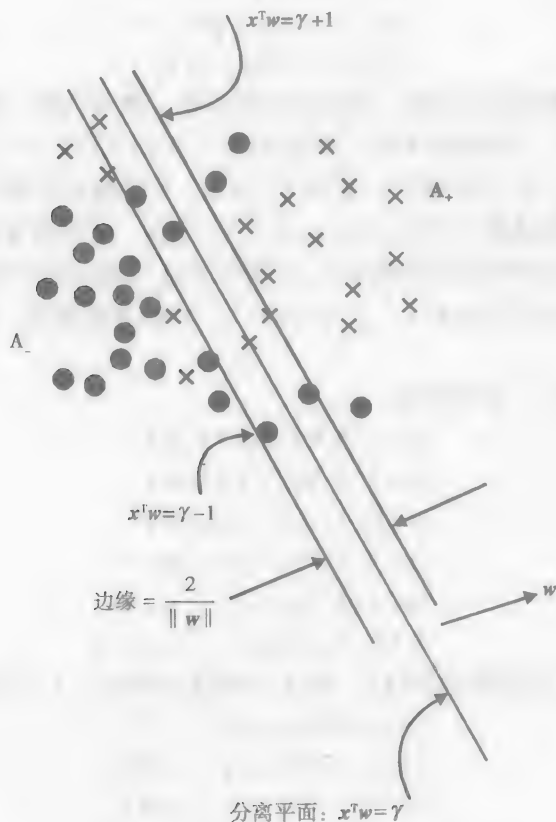


图 10-7 具有软边缘(边界)的 SVM

对于这个问题,使用线性核的标准 SVM 的公式以如下二次规划(参数 $\nu > 0$)给出:

$$\begin{aligned} \min_{(w, \gamma, y)} \quad & \nu e^T y + \frac{1}{2} w^T w \\ \text{满足} \quad & D(Aw - e\gamma) + y \geq e \\ & y \geq 0 \end{aligned} \quad (10.7)$$

其中 A 是一个 $m \times n$ 数据矩阵, e 是全为 1 的 $m \times 1$ 列向量, y 是 $m \times 1$ 误差向量。 D 是 $m \times n$ 对角阵, 其元素为 +1 或 -1 (+1 表示样本属于 A_+ , -1 表示样本属于 A_-)。 ν 是决定边缘和误差相对重要性的参数。

我们可以通过一个例子来解释这个过程。

设 A_+ 表示点集 (1, 0.8), (3, 2.5), (2.5, 1.8), A_- 表示点集 (1, 1.8), (3, 4.5), (2.5, 2.8)。

设 $w = (w_1, w_2)$ 和 γ 定义分隔这些点的超平面 $w_1 x_1 + w_2 x_2 - \gamma = 0$ 。如果假定这些点是线性可分的, 则设定约束使得任意点不能落于边界平面 $w_1 x_1 + w_2 x_2 - \gamma = 1$ 和 $w_1 x_1 + w_2 x_2 - \gamma = -1$ 定义的区间中。相应的, 我们有如下约束:

$$\begin{aligned} 1w_1 + 0.8w_2 - \gamma &\geq 1 \\ 3w_1 + 2.5w_2 - \gamma &\geq 1 \\ 2.5w_1 + 1w_2 - \gamma &\geq 1 \\ 1w_1 + 1.8w_2 - \gamma &\leq -1 \\ 3w_1 + 4.5w_2 - \gamma &\leq -1 \\ 2.5w_1 + 2.8w_2 - \gamma &\leq -1 \end{aligned}$$

如果并非所有点都是线性可分的, 则支持训练误差, 换句话说, 就是允许一些点落于边界超平面之间或越过另一个边界超平面。当类 A_+ 的一个点 $x_i(x_{i1}, x_{i2})$ 落于边界超平面之间或越过边界超平面(落于 A_- 的区域中, 即 $w^T x_i - \gamma \leq -1$ 界定的区域)时, 我们在不等式的左边加上一个正量 y_i , 使其满足约束 $w^T x_i - \gamma \geq +1$ 。现在, 我们有 $w^T x_i - \gamma + y_i \geq +1$ 。类似的, 对属于 A_- 类中落于边界超平面间或 A_+ 区域中的点, 我们在不等式左边减去一个正量 y_i 。对这样的点, 不等式表达为 $w^T x_i - \gamma - y_i \leq -1$ 。对所有其他点, 假定增加的量 y_i 的值为 0。

对于我们考察的例子, 其约束为:

$$\begin{aligned} 1w_1 + 0.8w_2 - \gamma + y_1 &\geq 1 \\ 3w_1 + 2.5w_2 - \gamma + y_2 &\geq 1 \\ 2.5w_1 + 1w_2 - \gamma + y_3 &\geq 1 \\ 1w_1 + 1.8w_2 - \gamma - y_4 &\leq -1 \\ 3w_1 + 4.5w_2 - \gamma - y_5 &\leq -1 \\ 2.5w_1 + 2.8w_2 - \gamma - y_6 &\leq -1 \end{aligned}$$

对 A_+ 中样本, 在不等式两边乘以 1, 对 A_- 中的样本乘以 -1, 我们有:

$$\begin{aligned} 1(w_1 + 0.8w_2 - \gamma) + y_1 &\geq 1 \\ 1(3w_1 + 2.5w_2 - \gamma) + y_2 &\geq 1 \\ 1(2.5w_1 + 1w_2 - \gamma) + y_3 &\geq 1 \\ -1(1w_1 + 1.8w_2 - \gamma) - y_4 &\geq 1 \\ -1(3w_1 + 4.5w_2 - \gamma) - y_5 &\geq 1 \end{aligned}$$

$$-1(2.5w_1 + 2.8w_2 - \gamma) - \gamma_6 \geq 1$$

用矩阵形式表达为:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} 1 & 0.8 \\ 3 & 2.5 \\ 2.5 & 1 \\ 1 & 1.8 \\ 3 & 4.5 \\ 2.5 & 2.8 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \gamma + \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \gamma_3 \\ \gamma_4 \\ \gamma_5 \\ \gamma_6 \end{bmatrix} \geq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

$$D(Aw - e\gamma) + y \geq e$$

$$y \geq 0$$

在式(10.7)中, n 是一个标量并且 $e^T y$ 是误差和。因此 $ve^T y$ 表示了所允许的误差的总量。相对于 w 和 γ 来最小化 $ve^T y + \frac{1}{2}w^T w$ 可以将边界平面间的距离最大化, 并且使越过对应边界平面的点数最小化。因为目标函数是二次的, 因此求解该问题需要二次规划。标准算法是 SMO 算法[9]。SVM^{light} (<http://svmlight.joachims.org/>) 是一个可用于求解上述 QP (Quadratic Programming, 二次规划) 问题的开源软件包。

如前所述, 解 w 可展开为:

$$w = \sum_{i=1}^k \alpha_i x_i$$

并且最终的决策函数为:

$$f(x) = \text{sign}(w^T x - \gamma) = \text{sign}\left(\sum_{i=1}^k \alpha_i (x_i^T x) - \gamma\right)$$

一旦我们确定了充当支持向量的数据点, 可以很容易求得 γ 。

在下一节中, 可以看到无需二次规划求解的 SVM 方法。该方法由 Mangasarian[11, 12] 提出。

10.3.1 线性 SVM 的线性规划公式表示

考察下面 SVM 公式表示

$$\begin{aligned} & \min_{(w, \gamma, y)} ve^T y + \frac{1}{2}w^T w \\ \text{满足} \quad & D(Aw - e\gamma) + y \geq e \\ & y \geq 0 \end{aligned} \quad (10.8)$$

Mangasarian[11, 12]证明下式

$$\begin{aligned} & \min_{(w, \gamma, y)} e^T y \\ \text{满足} \quad & D(Aw - e\gamma) + y \geq e \\ & y \geq 0 \end{aligned} \quad (10.9)$$

给出和原来公式几乎完全相同的结果。

新的公式等价于

$$\text{Min} \sum_{i=1}^k y_i$$

满足约束

$$\begin{aligned}
 w_1 x_{11} + w_1 x_{12} + \cdots - \gamma + y_1 &\geq 1 \\
 w_1 x_{21} + w_2 x_{22} + \cdots - \gamma + y_2 &\geq 1 \\
 &\vdots \\
 -1(w_1 x_{k1} + w_1 x_{k1} + \cdots - \gamma) + y_k &\geq 1
 \end{aligned}$$

该式可使用电子数据表求解器 (spreadsheet solver) 或 Matlab 轻松求解。式中的变量是 w 、 γ 和 y 。一旦通过求解上述线性规划问题得到 w 、 γ 的值, 分类器公式可写为:

$$f(x) = \text{sign}(w^T x - \gamma)$$

重要提示: 为了得到更好的分类结果, 在使用 SVM 前一定要规范化每个变量, 即使所有变量都在 $[0, 1]$ 或 $[-3, 3]$ 之间变化。

10.3.2 有训练误差的 SVM: 非线性核

式 (10.9) 所定义问题的非线性形式可由下式给出:

$$\min_{(u, \gamma, y)} e^T y$$

满足约束

$$\begin{aligned}
 D(K(A, A^T)u - e\gamma) + y &\geq e \\
 y &\geq 0
 \end{aligned} \quad (10.10)$$

其中, $K(A, A^T)$ 是非线性核函数。变量为向量 u 、 γ 和 y 。对于二阶多项式核, $K(A, A^T)$ 可以用 $(AA^T + 1)^2$ 计算。请注意, $(AA^T + 1)^2$ 的意思是对矩阵 AA^T 的每个元素加 1, 然后对矩阵的每个元素求平方。结果是一个 $m \times m$ 的矩阵, 其中 m 是数据点的个数。向量 u 的元素基本上是拉格朗日乘子。公式的原理在附录中给出。公式中的变量为向量 u 、 γ 和 y 。一旦通过求解线性规划问题得到 u 、 γ 的值, 分类器表达为:

$$f(x) = \text{sign}(K(x^T, A^T)u - \gamma)$$

10.4 邻近支持向量机

最近, 实现了一种更为简单的分类器, 邻近支持向量机 (Proximal Support Vector Machine, PSVM) [11, 12]。PSVM 将每个点归类于 (输入空间或特征空间中) 两个尽可能“推开”的平行平面中最近的一个。这种方法导致通过求解一个线性方程组, 产生线性或非线性分类器的快速、简单算法。

点的隔离 (式 (10.7) 中的优化问题) 被下面的问题替代:

$$\min_{(w, \gamma, y)} \nu \frac{1}{2} \|y\|^2 + \frac{1}{2} (w^T w + \gamma^2)$$

满足

$$D(Aw - e\gamma) + y = e \quad (10.11)$$

图 10-8 给出了该公式的几何解释。

对于 y 并没有非负的约束, 因为 y 现在表示点与穿过其所属类 (A_+ 或 A_-) 的数据簇中心的平面的偏差 (以 $1/\|w\|$ 度量) (见图 10-9)。请注意, 误差向量 y 的欧式范数而不是 $1 -$ 范数被最小化, 并且边界平面间的边缘相对于 w 的方向和 γ 与原点的相对位置而言被最大化了。

如 [11] 说明, 大量实践表明这种方法和式 (10.7) 定义的经典方法一样好, 并有更多的优势, 如目标函数的强凸性。该方法的核心思想是使计算简单, 但在式 (10.11) 中最基本的改变是不等式约束变成了等式约束。

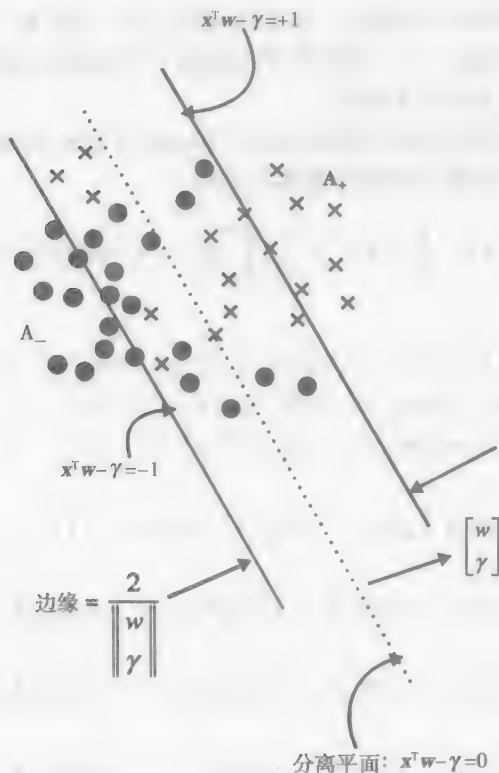
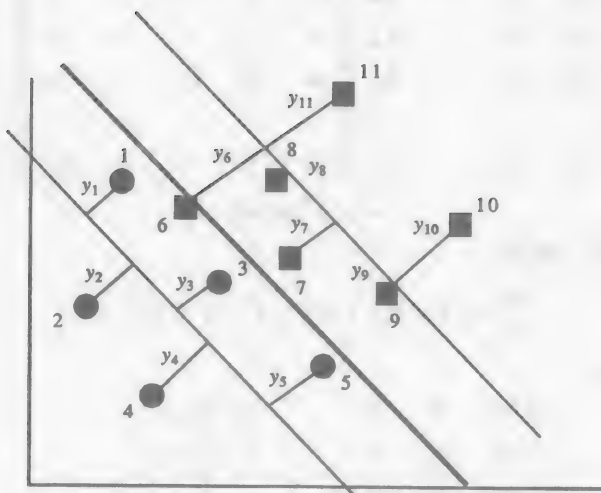


图 10-8 邻近支持向量分类

图 10-9 PSVM 中的 γ_i 的解释

即使是简单的修改也极大地改变了优化问题的本质。(现在)我们可以根据数据对问题明确给出准确解。由于其组合特点,这在以前的方法中是无法实现的。图 10-7 描绘了式(10.11)所定义的方法的几何意义,可以解释如下。平面 $x^T w - \gamma = \pm 1$ 不再是边界平面,而可以被认为是“邻近”平面,每个类的点在它们周围聚集,并且两个邻近平面的距离被目

标函数中的 $\mathbf{x}^T \mathbf{w} + \gamma^2$ 项分隔得尽可能远, 该项即是属于 R^{n+1} 的 (\mathbf{w}, γ) 空间中的两邻近平面的欧式范数距离的倒数。然而, 这一解释基于的思想并非是最大化边界(即两个平行边界平面间的距离)这一支持向量机的核心特征。

由式(10.11)所表达的等式约束问题的 KKT (Karush-Kuhn-Tucker) 充要优化条件可通过将关于 $(\mathbf{w}, \gamma, \mathbf{y}, \mathbf{u})$ 的拉格朗日的梯度置零而获得。

$$L(\mathbf{w}, \gamma, \mathbf{y}, \mathbf{u}) = \frac{\nu}{2} \|\mathbf{y}\|^2 + \frac{1}{2} \left\| \begin{bmatrix} \mathbf{w} \\ \gamma \end{bmatrix} \right\|^2 - \mathbf{u}^T [\mathbf{D}(\mathbf{A}\mathbf{w} - \mathbf{e}\gamma) + \mathbf{y} - \mathbf{e}] \quad (10.12)$$

其中 u_i 是拉格朗日乘子。

$$\begin{aligned} L = & (\nu/2) \cdot (\gamma_1^2 + \gamma_2^2 + \gamma_3^2 + \cdots + \gamma_m^2) + (1/2)(w_1^2 + w_2^2 + w_3^2 + \cdots + w_n^2 + \gamma^2) + \\ & -u_1(D_{11}(A_{11}w_1 + A_{12}w_2 + \cdots + A_{1n}w_n - \gamma) + y_1 - 1) \\ & -u_2(D_{22}(A_{21}w_1 + A_{22}w_2 + \cdots + A_{2n}w_n - \gamma) + y_2 - 1) \\ & \dots\dots\dots \\ & -u_m[D_{m1}(A_{m1}w_1 + A_{m2}w_2 + \cdots + A_{mn}w_n - \gamma) + y_1 - 1] \end{aligned}$$

对 w_1 求导, 得

$$\partial L / \partial w_1 = w_1 - u_1 D_{11} A_{11} - u_2 D_{22} A_{21} - \cdots - u_m D_{mm} A_{m1} = 0$$

类似地,

$$\partial L / \partial w_2 = w_2 - u_1 D_{11} A_{12} - u_2 D_{22} A_{22} - \cdots - u_m D_{mm} A_{m2} = 0$$

并且

$$\partial L / \partial w_n = w_n - u_1 D_{11} A_{1n} - u_2 D_{22} A_{2n} - \cdots - u_m D_{mm} A_{mn} = 0$$

这 m 个等式可以用向量形式表达为

$$\begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_n \end{bmatrix} - \begin{bmatrix} A_{11} & A_{21} & \cdots & A_{m1} \\ A_{12} & A_{22} & \cdots & A_{m2} \\ A_{13} & A_{23} & \cdots & A_{m3} \\ \vdots & \vdots & \ddots & \vdots \\ A_{1n} & A_{2n} & \cdots & A_{mn} \end{bmatrix} \begin{bmatrix} D_{11} & 0 & 0 & \cdots & 0 \\ 0 & D_{22} & 0 & \cdots & 0 \\ 0 & 0 & D_{33} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & D_{mm} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_m \end{bmatrix} = 0$$

$$\mathbf{w} - \mathbf{A}^T \mathbf{D} \mathbf{u} = 0 \quad (10.13)$$

对 γ 求导并令其等于零, 得到

$$\frac{\partial L}{\partial \gamma} = \gamma + u_1 D_{11} + u_2 D_{22} + \cdots + u_m D_{mm} = 0$$

用矩阵形式写为:

$$\gamma + (1 \quad 1 \quad 1 \quad \cdots \quad 1) \begin{bmatrix} D_{11} & 0 & 0 & \cdots & 0 \\ 0 & D_{22} & 0 & \cdots & 0 \\ 0 & 0 & D_{33} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & D_{mm} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_m \end{bmatrix} = 0$$

或

$$\gamma + \mathbf{e}^T \mathbf{D} \mathbf{u} = 0 \quad (10.14)$$

对 $\gamma_i (i=1, 2, \cdots, m)$ 求导, 并令其等于零, 得到:

$$\frac{\partial L}{\partial y_1} = \nu y_1 - u_1 = 0$$

$$\frac{\partial L}{\partial y_2} = \nu y_2 - u_2 = 0$$

.....

$$\frac{\partial L}{\partial y_m} = \nu y_m - u_m = 0$$

可以用矩阵形式表示为:

$$\nu \mathbf{y} - \mathbf{u} = 0 \quad (10.15)$$

最后, 对 u_1, u_2, \dots, u_m 求导并令其等于零, 得到 m 个等式, 其矩阵形式表达为

$$D(\mathbf{A}\mathbf{w} - \mathbf{e}\gamma) + \gamma - \mathbf{e} = 0 \quad (10.16)$$

收集所有方程, 得到 KKT 条件, 可以表达为:

$$\mathbf{w} - \mathbf{A}^T D\mathbf{u} = 0 \quad (10.17a)$$

$$\gamma + \mathbf{e}^T D\mathbf{u} = 0 \quad (10.17b)$$

$$\nu \mathbf{y} - \mathbf{u} = 0 \quad (10.17c)$$

$$D(\mathbf{A}\mathbf{w} - \mathbf{e}\gamma) + \gamma - \mathbf{e} = 0 \quad (10.17d)$$

方程(10.17)中的 a, b, c 给出了关于原始问题变量 $(\mathbf{w}, \gamma, \mathbf{y}, \mathbf{u})$ 关于拉格朗日乘子 \mathbf{u} 的以下表达:

$$\mathbf{w} = \mathbf{A}^T D\mathbf{u} \quad (10.18a)$$

$$\gamma = -\mathbf{e}^T D\mathbf{u} \quad (10.18b)$$

$$\mathbf{y} = \frac{\mathbf{u}}{\nu} \quad (10.18c)$$

将这些表达式代入式(10.17d)中, 得到 \mathbf{u} 在数据 \mathbf{A} 和 D 上的明确表达式如下:

从 KKT 条件, 得到

$$D(\mathbf{A}\mathbf{w} - \mathbf{e}\gamma) + \gamma - \mathbf{e} = 0 \quad (10.19)$$

将 \mathbf{w}, γ 和 \mathbf{y} 代入公式(10.19), 得到

$$D(\mathbf{A}\mathbf{A}^T D\mathbf{u} - \mathbf{e}\mathbf{e}^T D\mathbf{u}) + \frac{\mathbf{u}}{\nu} = \mathbf{e}$$

$$[D(\mathbf{A}\mathbf{A}^T - \mathbf{e}\mathbf{e}^T)D]\mathbf{u} + \frac{\mathbf{u}}{\nu} = \mathbf{e}$$

$$[D(\mathbf{A}\mathbf{A}^T - \mathbf{e}\mathbf{e}^T)D + \frac{I}{\nu}]\mathbf{u} = \mathbf{e}$$

$$\mathbf{u} = \left(\frac{I}{\nu} + D(\mathbf{A}\mathbf{A}^T - \mathbf{e}\mathbf{e}^T)D \right)^{-1} \mathbf{e} = \left(\frac{I}{\nu} + \mathbf{H}\mathbf{H}^T \right)^{-1} \mathbf{e} \quad (10.20)$$

其中 \mathbf{H} 定义为:

$$\mathbf{H} = D[\mathbf{A} - \mathbf{e}] \quad (10.21)$$

因此有

$$\mathbf{H} = D[\mathbf{A} - \mathbf{e}] \Rightarrow \mathbf{H}\mathbf{H}^T = D[\mathbf{A} - \mathbf{e}] \begin{bmatrix} \mathbf{A} \\ -\mathbf{e}^T \end{bmatrix} D^T$$

$$= D[\mathbf{A}\mathbf{A}^T + \mathbf{e}\mathbf{e}^T]D^T$$

$$= D[\mathbf{A}\mathbf{A}^T + \mathbf{e}\mathbf{e}^T]D \quad (\text{因为 } D \text{ 是对称的})$$

因此有

$$u = \left(\frac{1}{n} + D(AA^T - ee^T)D \right)^{-1} e = \left(\frac{I}{n} + HH^T \right)^{-1} e \quad (10.22)$$

从式(10.22)中得到 u , 式(10.18)给出了式(10.11)定义的问题的明确解答 (w , y , γ)。由于式(10.21)给出 u 的解需要对一个可能很大的 $m \times m$ 矩阵求逆, 因此我们使用 Sherman-Morrison-Woodbury 公式对矩阵求逆 (见文献[11]), 得到:

$$u = v \left(I - H \left(\frac{I}{v} + H^T H \right)^{-1} H^T \right) e \quad (10.23)$$

这一公式只需对 $(I/v + H^T H)$ 求逆, 其维数只有 $(n+1) \times (n+1)$ 。注意, n 是预测变量的个数, 在大多数实际情况下, 其值小于 100。一旦得到 u 的值, w 和 γ 就可以从式(10.18)中方便地计算出来。图 10-10 给出了实现 PSVM 的 Matlab 代码。

```
function[w,gamma]= psvm(A,D,nu)
% psvm linear & nonlinear classification
% input:A,D,nu. output:w,gamma
% [w,gamma]=psvm(A,D,nu)
[m,n]=size(A);e=ones(m,1);H=D*[A -e];
r=sum(H)'; %r=H'*e;
r=(speye(n+1)/nu+H'*H)\r; %solve(I/nu+H'*H)r=H'*e
u=nu*(1-(H*r));s=D*u;
w=(s'*A)'; %w=A'*D*u
gamma=-sum(s); %gamma=-e'*D*u
x=A';
z=sign(w'*x-gamma)
```

图 10-10 线性 PSVM 的 Matlab 代码

例 10.1 假定 A_+ 中的数据为: (0, 0), (3, 4), (5, 9), (12, 1), (8, 7)。 A_- 中的数据为(9, 8), (6, 12), (10, 8), (8, 5)(14, 8)。表 10-1 以表格形式给出了这些数据。

对变量 x_1 和 x_2 进行规范化, 数据列于表 10-2。

表 10-1 用于寻找 SVM 分类器的数据

x_1	x_2	Class
0	0	1
3	4	1
5	9	1
12	1	1
8	7	1
9	8	-1
6	12	-1
10	8	-1
8	5	-1
14	8	-1

表 10-2 用于 SVM 分类的规范化数据

-1.7984	-1.6730	1
-1.0791	-0.5937	1
-0.5995	0.7556	1
1.0791	-1.4032	1
0.1199	0.2159	1
0.3597	0.4857	-1
-0.3597	1.5651	-1
0.5995	0.4857	-1
0.1199	-0.3238	-1
1.55876	0.4857	-1

现在计算增广矩阵 $[A - e]$:

$$[A - e] = \begin{bmatrix} -1.7984 & -1.6730 & -1 \\ -1.0791 & -0.5937 & -1 \\ -0.5995 & 0.7556 & -1 \\ 1.0791 & -1.4032 & -1 \\ 0.1199 & 0.2159 & -1 \\ 0.3597 & 0.4857 & -1 \\ -0.3597 & 1.5651 & -1 \\ 0.5995 & 0.4857 & -1 \\ 0.1199 & -0.3238 & -1 \\ 1.5588 & 0.4857 & -1 \end{bmatrix}$$

计算 $H = D[A - e]$ 和 $H^T H$:

$$H = D[A - e] = \begin{bmatrix} -1.7984 & -1.6730 & -1 \\ -1.0791 & -0.5937 & -1 \\ -0.5995 & 0.7556 & -1 \\ 1.0791 & -1.4032 & -1 \\ 0.1199 & 0.2159 & -1 \\ -0.3597 & -0.4857 & 1 \\ 0.3597 & -1.5651 & 1 \\ -0.5995 & -0.4857 & 1 \\ -0.1199 & 0.3238 & 1 \\ -1.5588 & -0.4857 & 1 \end{bmatrix}$$

$$H^T H = \begin{bmatrix} 9 & 2.33 & 0 \\ 2.33 & 9 & 0 \\ 0 & 0 & 10 \end{bmatrix}$$

现在, 我们进一步计算 $[I - H(\frac{I}{\nu} + H^T H)^{-1} H^T]$:

$$I/\nu = \begin{bmatrix} 1/\nu & 0 & 0 \\ 0 & 1/\nu & 0 \\ 0 & 0 & 1/\nu \end{bmatrix}$$

$$I/\nu (\text{取 } \nu = 0.1) = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 10 \end{bmatrix}$$

$$\left(\frac{I}{\nu} + H^T H\right) = \begin{bmatrix} 19 & 2.33 & 0 \\ 2.33 & 19 & 0 \\ 0 & 0 & 20 \end{bmatrix}$$

$$\left(\frac{I}{\nu} + H^T H\right)^{-1} = \begin{bmatrix} 19 & 2.33 & 0 \\ 2.33 & 19 & 0 \\ 0 & 0 & 20 \end{bmatrix}^{-1}$$

$$= \begin{bmatrix} 0.0534 & -0.0066 & 0 \\ -0.0066 & 0.0534 & 0 \\ 0 & 0 & 0.05 \end{bmatrix}$$

$$\left[I - H \left(\frac{I}{\nu} + H^T H \right)^{-1} H^T \right] =$$

$$\begin{bmatrix} 0.667 & -0.188 & -0.042 & -0.067 & -0.023 & -0.018 & -0.041 & -0.039 & 0.065 & -0.120 \\ -0.188 & 0.877 & -0.064 & -0.027 & -0.038 & 0.019 & 0.031 & 0.006 & 0.052 & -0.046 \\ -0.042 & -0.064 & 0.894 & 0.052 & -0.055 & 0.058 & 0.133 & 0.049 & 0.031 & 0.014 \\ -0.067 & -0.027 & 0.052 & 0.763 & -0.040 & 0.034 & -0.102 & 0.050 & 0.085 & 0.114 \\ -0.023 & -0.038 & -0.055 & -0.040 & 0.947 & 0.057 & 0.065 & 0.058 & 0.047 & 0.063 \\ -0.018 & 0.019 & 0.058 & 0.034 & 0.057 & 0.933 & -0.081 & -0.071 & -0.044 & -0.086 \\ -0.041 & 0.031 & 0.133 & -0.102 & 0.065 & -0.081 & 0.805 & -0.074 & -0.019 & -0.046 \\ -0.039 & 0.006 & 0.049 & 0.050 & 0.058 & -0.071 & -0.074 & 0.922 & -0.046 & -0.106 \\ 0.065 & 0.052 & 0.031 & 0.085 & 0.047 & -0.044 & -0.019 & -0.046 & 0.943 & -0.055 \\ -0.120 & -0.046 & 0.014 & 0.114 & 0.063 & -0.086 & -0.046 & -0.106 & -0.055 & 0.817 \end{bmatrix}$$

$$u = \nu \left(I - H \left(\frac{I}{\nu} + H^T H \right)^{-1} H^T \right) e =$$

$$\begin{bmatrix} 0.0193 \\ 0.0622 \\ 0.1071 \\ 0.0862 \\ 0.1081 \\ 0.0800 \\ 0.0670 \\ 0.0750 \\ 0.1059 \\ 0.0550 \end{bmatrix}$$

$$w = A^T D u = (-0.2081, -0.2585)$$

$$\gamma = -e^T D u = -2.0817e - 0.17 = 0$$

表 10-3 给出了分类结果。

表 10-3 SVM 训练的结果

x_1	x_2	$w^T x - \gamma$	class
-1.7984	-1.6730	0.8068	1
-1.0791	-0.5937	0.3781	1
-0.5995	0.7556	-0.0706	-1
1.0791	-1.4032	0.1382	1
0.1199	0.2159	-0.0808	-1
0.3597	0.4857	-0.2004	-1
-0.3597	1.5651	-0.3298	-1
0.5995	0.4857	-0.2503	-1
0.1199	-0.3238	0.0588	1
1.5588	0.4857	-0.4500	-1

可以看到，有三个数据被误分类了。它们是第 3 个、第 5 个和第 9 个数据点。在训练数据集上的分类准确率是 70%。

使用非线性核

为了得到非线性分类器,我们修改等式约束优化问题,将原变量 w 用其对偶等价 $w = A^T Du$ 替换,得到

$$\min_{(u, \gamma, y)} \nu \frac{1}{2} \|y\|^2 + \frac{1}{2} (u^T u + \gamma^2)$$

$$\text{满足} \quad D(AA^T Du - e\gamma) + y = e$$

其中,目标函数必须被修改为最小化变量 (u, γ, y) 的加权欧式范数之和。

注意,在目标函数中 $w^T w$ 被以 $u^T u$ 替代,因为最小化 $w^T w$ 相当于最小化 $u^T u$ 。在约束中, w 被替代以 $A^T Du$ 。

如果我们将线性核 AA^T 替换为非线性核 $k(A, A^T)$, 我们得到:

$$\min_{(u, \gamma, y)} \nu \frac{1}{2} \|y\|^2 + \frac{1}{2} (u^T u + \gamma^2)$$

满足约束

$$D[k(A, A^T)Du - e\gamma] + y = e$$

使用简写

$$K = K(A, A^T)$$

其拉格朗日方程为

$$L(u, \gamma, y, t) = \left(\frac{1}{2}\right)\nu \|y\|^2 + \frac{1}{2}(uu^T + \gamma^2) + t[D(KDu - e\gamma) + y - e]$$

这里, $t \in R^m$ 是与等式约束相关的拉格朗日乘子。令关于 (u, γ, y, t) 的梯度为零,得到下面的 KKT 优化条件:

$$u - DK^T Dt = 0 \quad (10.24a)$$

$$\gamma + e^T Dt = 0 \quad (10.24b)$$

$$\nu y - t = 0 \quad (10.24c)$$

$$D(KDu - e\gamma) + y = e \quad (10.24d)$$

式 10.24a、10.24b 和 10.24c 给出了 (u, γ, y) 关于拉格朗日乘子 t 的表达式:

$$u = DK^T Dt, \quad \gamma = -e^T Dt, \quad y = \frac{t}{\nu} \quad (10.25)$$

将其代到等式中,得到了 t 在数据 A 和 D 上的表达式:

$$t = \left(\left(\frac{I}{\nu} \right) + D(KK^T + ee^T)D \right)^{-1} e = \left(\left(\frac{I}{\nu} \right) + GG^T \right)^{-1} e \quad (10.26)$$

其中 G 定义为:

$$G = D[K - e]$$

G 和 H 的定义中有一个相似之处。这一相似使得我们可以用 K 替代 A 而由 H 得到 G 。这样,在图 10-11 的算法中用 K 替代 A 就得到一个非线性分类器。图 10-11 给出了非线性 SVM 的 Matlab 的代码。

这里 $K = (AA^T + 1)^4$ 。

$(AA^T + 1)^4$ 的含义是对矩阵 AA^T 的每个元素加 1, 然后求它的四次方。

```

function [t,gamma,k] = psvm(k,D,nu)
% PSVM:linear and nonlinear classification
% INPUT : A,D,nu. OUTPUT : t,gamma
% [t,gamma]=psvm(A,D,nu);
[m,n]=size(k);
e=ones(m,1);
G=D*[k -e];
r=sum(G)'; %r=G'*e;
r=(speye(n+1)/nu+G'*G)\r; %solve (I/nu+G'*G)r=G'*e
u=nu*(1-(G*r))
s=D*u;
t=(s'*k)'; %t=K'*D*u
gamma=-sum(s); %gamma=-e'*D*u
b=sign((t'*k)-gamma)

```

图 10-11 非线性 SVM 的 Matlab 代码

例 10.2 对于例 10.1, 我们使用一个多项式核 $K(x, y) = (xy^T + 1)^4$ 。将其应用于整个数据向量, 得到

$$K(A, A^T) = (AA^T + 1)^4$$

$(AA^T + 1)^4$ 的含义是对矩阵 AA^T 的每个元素加 1, 然后求它的四次方。

取 $v = 0.3$, 得到 u 为 $(-0.0011, 0.0128, -0.0109, 0.0043, 0.3735, 0.1184, -0.0023, 0.0531, 0.2365, -0.0090)$, γ 为 0.0182。

对一个新的数据向量 x , 分类可通过取 $K(x^T, A^T)u - \gamma$ 的符号获得。下面我们通过两个财务方面的例子来解释其有用性。

例 10.3 一个银行的商业贷款部门的经理想要建立一个规则来决定是否批准各种贷款请求。经理认为在做决策时, 一个公司业绩的三个关键特征是非常重要的: 资产折现力 (liquidity)、盈利能力 (profitability) 和活性 (activity)。这个经理以目前资产和负债比作为资产折现力的度量。盈利能力以净利润和销售额之比度量。活性以销售额和固定资产之比度量。这个经理收集了银行在过去五年内所做贷款的 18 个样本列于表 10-4。这些贷款被分为两组: ①被批准的贷款; ②被拒绝的贷款。

表 10-4 公司的业绩数据

类别	资产折现力	盈利能力	活性
1	0.9	0.34	1.53
1	0.88	0.23	1.67
1	0.92	0.28	1.43
1	0.89	0.14	1.24
1	0.78	0.35	1.8
1	0.81	0.26	2.01
1	0.72	0.18	1.75
1	0.93	0.22	0.99
1	0.82	0.26	1.4
-1	0.78	0.26	1.34
-1	0.78	0.27	1.67
-1	0.72	0.18	1.53
-1	0.69	0.16	1.2

(续)

类别	资产折现力	盈利能力	活性
-1	0.63	0.15	0.88
-1	0.58	0.22	1.42
-1	0.81	0.18	1.59
-1	0.67	0.21	1.21
-1	0.65	0.16	1.37

表 10-5 列出了规范化后的数据。

表 10-5 规范化数据

x_1	x_2	x_3	类
1.18	1.86	0.29	1
0.99	0.08	0.79	1
1.37	0.89	-0.06	1
1.09	-1.37	-0.72	1
0.04	2.02	1.24	1
0.33	0.56	1.98	1
-0.53	-0.73	1.07	1
1.47	-0.08	-1.6	1
0.42	0.56	-0.16	1
0.04	0.56	-0.37	-1
0.04	0.73	0.79	-1
-0.53	-0.73	0.29	-1
-0.81	-1.05	-0.86	-1
-1.38	-1.21	-1.99	-1
-1.86	-0.08	-0.09	-1
0.33	-0.73	0.51	-1
-1	-0.24	-0.83	-1
-1.19	-1.05	-0.27	-1

通过训练线性支持向量机,得到下面的权重向量:

$$w = (0.4236, 0.1312, 0.1268), \gamma = 4.1633e - 0.17$$

表 10-6 为每个数据给出其 $w^T x - \gamma$ 值。 $w^T x - \gamma$ 的正负决定了其分类。

表 10-6 每个数据的 $w^T x - \gamma$ 值和分类

x_1	x_2	x_3	$w^T x - \gamma$	类	正确分类/错误分类
1.181975	1.856231	0.294646	0.7816	1	Yes
0.992015	0.080706	0.786372	0.5305	1	Yes
1.371935	0.887763	-0.05659	0.6905	1	Yes
1.086995	-1.372	-0.72393	0.1886	1	Yes
0.042213	2.017642	1.242975	0.4402	1	Yes
0.327154	0.56494	1.980565	0.4638	1	Yes
-0.52767	-0.72635	1.067358	-0.1835	-1	No
1.466915	-0.08071	-1.60201	0.4077	1	Yes
0.422134	0.56494	-0.16196	0.2324	1	Yes
0.042213	0.56494	-0.3727	0.0447	1	No
0.042213	0.726351	0.786372	0.2129	1	No
-0.52767	-0.72635	0.294646	-0.2815	-1	Yes

(续)

x_1	x_2	x_3	$w^T x - \gamma$	类	正确分类/错误分类
-0.81261	-1.04917	-0.86442	-0.5915	-1	Yes
-1.38249	-1.21059	-1.98837	-0.9966	-1	Yes
-1.85739	-0.08071	-0.09171	-0.809	-1	Yes
0.327154	-0.72635	0.505385	0.1074	1	No
-1.00257	-0.24212	-0.8293	-0.5616	-1	Yes
-1.19253	-1.04917	-0.26733	-0.6767	-1	Yes

因为数据经过规范化, 因此分隔超平面为:

$$\frac{w_1(x_1 - \bar{x}_1)}{\sigma_1 + w_2} + \frac{w_2(x_2 - \bar{x}_2)}{\sigma_2 + w_3} + \frac{w_3(x_3 - \bar{x}_3)}{s_3 - \gamma} - \gamma = 0$$

$$\frac{0.4236(x_1 - 0.7756)}{0.1053} + \frac{0.1312(x_2 - 0.225)}{0.06195} + \frac{0.1268(x_3 - 1.4461)}{0.2847} - \gamma = 0$$

线性核(在训练数据上的)分类正确率为 78%。如果使用四阶的多项式核, 分类正确率为 100%。

例 10.4 保险公司 HDIC 希望开发一个程序来帮助预测一个银行是否会在来年陷入财务危机。以下财务比率对做这样的预测是有帮助的。

比率 1 = 总资金/总资产

比率 2 = 总开销/总资产

比率 3 = 总借贷/总储蓄

HDIC 针对银行收集了一年的数据样本。数据中包括了目前正陷入财务危机的银行。这些数据汇总在表 10-7 中。

表 10-7 银行及其财务比率

对象	分组	比率 1	比率 2	比率 3
1	1	1	1	8.1
2	1	6.6	0.1	1.04
3	1	5.8	0.11	0.66
4	1	12.3	0.09	0.8
5	1	4.5	0.11	0.69
6	1	9.1	0.14	0.74
7	1	1.1	0.12	0.63
8	1	8.9	0.12	0.75
9	1	0.7	0.16	0.56
10	1	9.8	0.12	0.65
11	2	7.3	0.1	0.55
12	2	14	0.08	0.46
13	2	9.6	0.08	0.72
14	2	12.4	0.08	0.43
15	2	18.4	0.07	0.52
16	2	8	0.08	0.54
17	2	12.6	0.09	0.3
18	2	9.8	0.07	0.67
19	2	8.3	0.09	0.51

分组 1 中的银行是陷入财务危机的银行, 而分组 2 中的银行是财务状况良好的银行。

规范化后的数据如表 10-8 所示。

表 10-8 规范化财务比率

x_1	x_2	x_3	类
-1.61696	4.102026	4.111626	1
-0.39851	-0.23056	0.013443	1
-0.57258	-0.18242	-0.20714	1
0.84169	-0.2787	-0.12587	1
-0.85543	-0.18242	-0.18973	1
0.145435	-0.03801	-0.1607	1
-1.5952	-0.13428	-0.22455	1
0.101919	-0.13428	-0.1549	1
-1.68223	0.058275	-0.26519	1
0.297741	-0.13428	-0.21294	1
-0.24621	-0.23056	-0.27099	-1
1.211575	-0.32684	-0.32324	-1
0.254225	-0.32684	-0.17231	-1
0.863447	-0.32684	-0.34065	-1
2.168925	-0.37498	-0.28841	-1
-0.0939	-0.32684	-0.2768	-1
0.906963	-0.2787	-0.41611	-1
0.297741	-0.37498	-0.20133	-1
-0.02863	-0.2787	-0.29421	-1

通过训练,得到 w 向量为 $(-0.4663, 0.0640, 0.0255)$ 以及 $\gamma = -0.0448$ 。

表 10-9 给出了每个数据的 $w^T x - \gamma$ 值。 $w^T x - \gamma$ 的正负决定了分类。

表 10-9 (规范化) 银行危机数据的 $w^T x - \gamma$ 值和分类

x_1	x_2	x_3	$w^T x - \gamma$	类	正确分类/错误分类
-1.61696	4.102026	4.111626	1.166	1	Yes
-0.39851	-0.23056	0.013443	0.2162	1	Yes
-0.57258	-0.18242	-0.20714	0.2948	1	Yes
0.84169	-0.2787	-0.12587	-0.3687	-1	No
-0.85543	-0.18242	-0.18973	0.4271	1	Yes
0.145435	-0.03801	-0.1607	-0.0296	-1	No
-1.5952	-0.13428	-0.22455	0.7742	1	Yes
0.101919	-0.13428	-0.1549	-0.0153	-1	No
-1.68223	0.058275	-0.26519	0.8261	1	Yes
0.297741	-0.13428	-0.21294	-0.1081	-1	Yes
-0.24621	-0.23056	-0.27099	0.1379	1	No
1.211575	-0.32684	-0.32324	-0.5493	-1	Yes
0.254225	-0.32684	-0.17231	-0.0991	-1	Yes
0.863447	-0.32684	-0.34065	-0.3874	-1	Yes
2.168925	-0.37498	-0.28841	-0.9978	-1	Yes
-0.0939	-0.32684	-0.2768	0.0606	1	No
0.906963	-0.2787	-0.41611	-0.4065	-1	Yes
0.297741	-0.37498	-0.20133	-0.1232	-1	Yes
-0.02863	-0.2787	-0.29421	-0.0328	1	No

使用线性核的分类正确率为 63%。使用四阶的多项式核,分类正确率为 100%。

10.5 生成数据集

现在已经有了各种 SVM 方法并且应用特定核的新的 SVM 也在文献中不断出现。当设计一个新的算法的时候,我们应该和其他已有的算法进行比较[3, 7, 8]。为了比较 SVM 的性能,研究者通常使用一些大学数据库中公共标准数据集。另外,研究者也使用人工生成的数据集,即使是使用许多标准的非线性核,这些数据也很难被完全分类。这里我们讨论三种人工数据集生成器,前两种是二维的(二元),而最后一种是 n 维的。

10.5.1 螺旋数据生成器

我们以半径 6.5 生成两个缠绕的螺旋。这两个螺旋圈(内向)互相缠绕三圈,每个螺旋生成 97 个样本。两个相继螺旋点之间的角度为 $\pi/16$ 。图 10-12 给出了生成这些点的 C 语言代码。这些点的一个集合被标记为“+1”,其他集合标记为“-1”。目标是进行学习以区分这两个类。图 10-13 给出了生成数据点的图示。

```
main( )
{
    int i;
    double x, y, angle, radius;

    /* write spiral of data */
    for (i=0; i<=96; i++) {
        angle = i * PI / 16.0;
        radius = 6.5 * (104 - i) / 104.0;
        x = radius * sin(angle);
        y = radius * cos(angle);
        printf("(%.5f %.5f) (%3.1f)\n", x, y, 1.0);
        printf("(%.5f %.5f) (%3.1f)\n", -x, -y, -1.0);
    }
    //PI=3.14
```

图 10-12 生成螺旋数据集的 C 代码

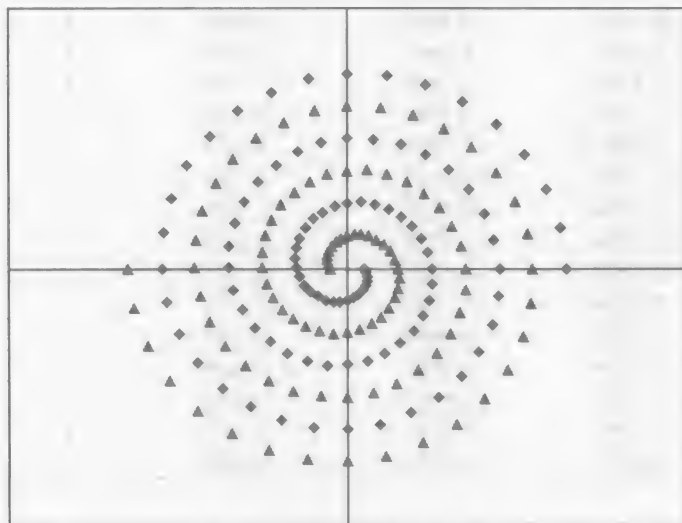


图 10-13 两个螺旋数据集

10.5.2 棋盘格数据集

这里, 我们生成两个数据点集。一个集合构成图 10-14 中的黑方框, 另一个集合构成白方框。以 (x_L, x_U) 和 (y_L, y_U) 分别表示方框的 x, y 坐标的下限和上限。

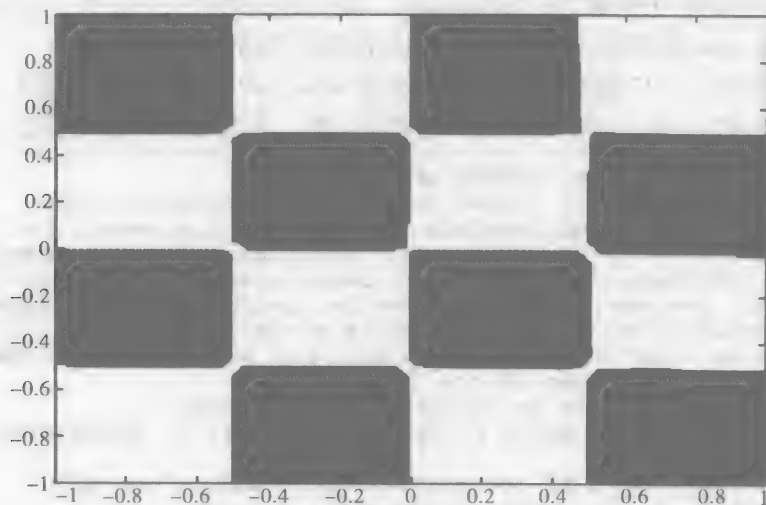


图 10-14 棋盘

例如, 对于左上角的黑方框, 其上下限分别为:

$$x_L = -1, x_U = -0.5$$

$$y_L = 0.5, y_U = 1$$

从一个方框中随机选择的点的坐标如下:

$$x = x_L + (x_U - x_L) * \text{rand}()$$

$$y = y_L + (y_U - y_L) * \text{rand}()$$

其中 $\text{rand}()$ 是区间 $0 \sim 1$ 之间的均匀随机数据生成器。图 10-15 给出了生成的数据点。

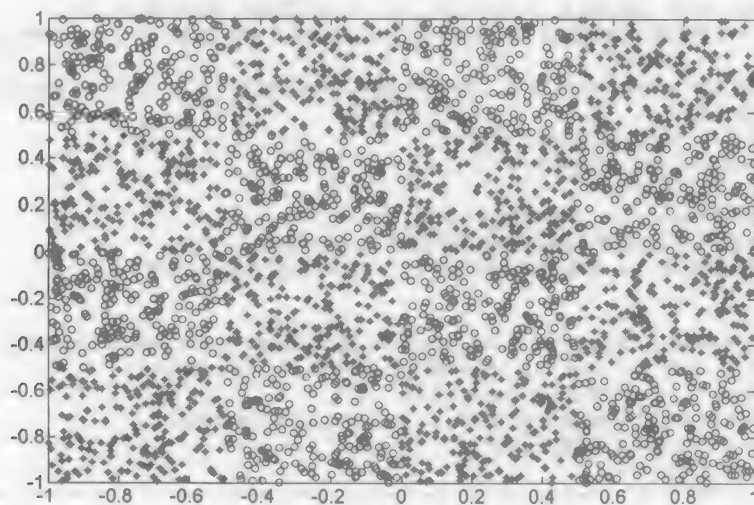


图 10-15 从棋盘产生的数据

10.5.3 多元正态分布数据生成器

在以上两个数据集生成器中,我们限制在二维数据点。O. L Mangasarian [11, 12] 给出了生成多元正态分布数据集的 Matlab 代码。代码可从 <http://www.cs.wisc.edu/dmi/svm/ndc/> 下载。代码的商业化使用需要作者的授权。我们将代码列于此处用于参考(见图 10-16, 感谢 O. L Mangasarian 教授允许我们在本书中使用这些代码)。

代码产生 20 000 个 32 维的数据(分别由变量 nRows 和 nCols 控制)。落入“+1”和“-1”分组的点的个数依赖于由代码随机生成的超平面边界,因而无法由用户设定。

```
% NORMALLY DISTRIBUTED CLUSTERS data generator.
% Generate a series of random centers for multivariate normal
% distributions. Randomly generate a fraction for this center, i.e. what
% fraction of points we will get from this center. Randomly generate a
% separating plane. Based on plane, choose classes for centers. Then
% randomly generate the points from the distributions. Can increase
% inseparability by increasng variances of distributions. We will get
% measure of "true" separability by looking at how many points ended up
% on the opposite sides of the line.
% All values are taken as integers for simplicity.
% Copyright (C) 2000 David R. Musicant and Olvi L. Mangasarian.
% Version 1.0
% This software is free for academic and research use only.
% For commercial use, contact musicant@cs.wisc.edu.

rand('state',91225);
randn('state',19481);
lowBound = -50;
highBound = 50;
nCenters = 20;
nCols = 32;
nRows = 20000;
nTestRows = 0.01 * nRows;
nBufferPoints = 100000;
nExpandFactor = 10; % How much to stretch the covariance matrix
sTrainFile = 'outtrain.txt';
sTestFile = 'outtest.txt';

% Generate the centers according to a uniform distribution.
mCenters = round(lowBound + rand(nCenters,nCols)*(highBound-lowBound));

% Generate the variances and covariances randomly to create a matrix for
% each center
mCovariance = zeros(nCols,nCols);
cCovariance = cell(nCenters,1);
for i = 1:nCenters,
    mRootCovariance = nExpandFactor * ...
        rand(nCols,nCols)*(highBound-lowBound) / 50;
    cCovariance{i} = mRootCovariance' * mRootCovariance;
end;

% Determine what proportion of points will come from each center, then
% create a cdf to use in deciding which to generate.
vPointFraction = rand(nCenters,1);
vPointFraction = vPointFraction / sum(vPointFraction);
vPointCdf = zeros(1,nCenters);
for i = 1:nCenters,
```

图 10-16 生成多元数据的 Matlab 代码

```

    vPointCdf(i) = sum(vPointFraction(1:i));
end;

% Create a random separating plane.
w = -2 + rand(nCols,1)*4;
gamma = lowBound / 10 + rand * (highBound-lowBound)/10;

% Now choose which classes to which each center belongs
vCenterClasses = sign(mCenters * w - gamma * ones(nCenters,1));
vZeroSpots = find(vCenterClasses==0);
vCenterClasses(vZeroSpots) = ones(length(vZeroSpots),1);

% Prepare output file
flatfile([],sTrainFile,0);
flatfile([],sTestFile,0);

% Now go through and begin generating random points.
% Do it twice: once for testing, once for training.

for nDataset = 1:2,

    if (nDataset==1)
        nRowsLeft = nRows;
        sOutputFile = sTrainFile;
        nTotRows = nRows;
    else
        nRowsLeft = nTestRows;
        sOutputFile = sTestFile;
        nTotRows = nTestRows;
    end;
    nMisclass = 0;
    nTrainingClass1 = 0;
    nTrainingClassm1 = 0;

    while (nRowsLeft > 0)
        disp(sprintf('Rows left = %d',nRowsLeft));
        nRowsNow = min(nBufferPoints,nRowsLeft);
        nRowsLeft = nRowsLeft - nRowsNow;
        mNewPoints = zeros(nRowsNow,nCols);
        vPointCenters = zeros(nRowsNow,1);

        % Determine which center each point should belong to
        vRandomNumbers = rand(nRowsNow,1);
        for i = nCenters:-1:1,
            vCenterMatch = (vRandomNumbers <= vPointCdf(i));
            vPointCenters([vCenterMatch]) = i;
        end;

        % Create a vector of training classes for each point
        vTrainingClasses = zeros(nRowsNow,1);

        % Within each class, generate an appropriate number of random points.
        for i = 1:nCenters,
            vIndices = (vPointCenters==i);
            nPoints = sum(vIndices);
            vTrainingClasses(vIndices) = vCenterClasses(i);
            mNewPoints(vIndices,:) = round( ...
                mvnrnd(mCenters(i,:),cCovariance{i},nPoints));
        end;
    end;
end;

```

图 10-16 (续)

```

% Count how many points are incorrectly classified
vFitClass = sign(mNewPoints(vIndices,:) * w - gamma * ...
ones(nPoints,1));
vZeroSpots = find(vFitClass==0);
vFitClass(vZeroSpots) = ones(length(vZeroSpots),1);
nMisclass = nMisclass + sum(vFitClass~=vCenterClasses(i));
end; %for

% Output the data points to disk
flatfile([mNewPoints vTrainingClasses],sOutputFile,1);
nTrainingClass1 = nTrainingClass1 + sum(vTrainingClasses==1);
n+TrainingClassm1 = nTrainingClassm1 + sum(vTrainingClasses== -1);

end; %while

disp(sprintf('Percent separable estimate = %4.2f%%\n',100*(1-
nMisclass/ nTotRows)));
disp(sprintf('Number class 1 points = %d\n',nTrainingClass1));
disp(sprintf('Number class -1 points = %d\n',nTrainingClassm1));

end; %for-nDataset

```

图 10-16 (续)

10.6 问题及解答

在这一节中，我们给出一些常见问题(任何认真学习 SVM 的人都有可能遇到这些问题)。如果你在阅读以前各节时提出了这些问题，显然你已经在正确的轨道上。

1) 改变核是否会改变决策边界的位置?

答：多数情况下是这样的。某些时候，改变径向基核的 σ 值，并不改变决策边界的位置，如，数据点是对称的。

2) 正数据点和负数据点之间的距离会对分隔区间的宽度以及支持向量的权重产生怎样的影响?

答：最邻近的异类点之间的距离越大，分隔区间的宽度越大，同时形成分隔区间的支持向量的权重越小。直觉上，支持向量的距离越大，它们对落于分隔区间的节点的“影响”，即“吸引”这些点到其分类中的影响力越小，权重越低。见图 10-17。

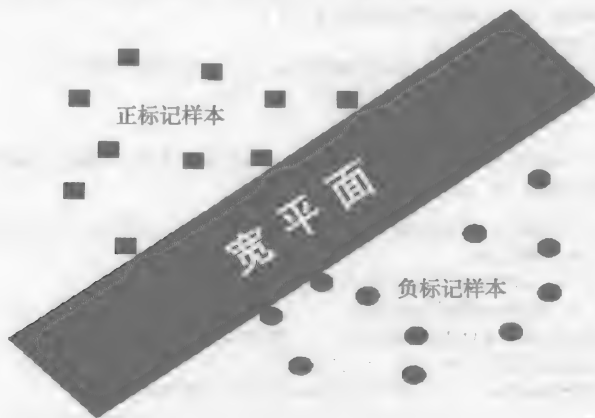


图 10-17 平面越宽则支持向量的权重越小

3) 对于一个在数据上过分拟合的支持向量机, 支持向量的个数有何影响?

答: 大多数或全部数据点都会是支持向量。

4) 对某个数据集, 如何通过比较图表来判断哪个核构建的分类器更好?

答: 对于一个训练数据集, 最好的分类器是将全部数据点正确分类。同时, 并非全部 (或并非全部而是少部分) 数据点是支持向量。

习题

1. 线性支持向量机使用什么方程分类?
2. 在使用线性支持向量机决定决策边界的位置, 问题用拉格朗日形式表示时, 需要最大化什么函数? 如何表达约束?
3. 线性支持向量机在分类器内部使用点积。在训练阶段是否也使用点积? 如果是, 在哪里使用?
4. 最好的决策边界产生最宽的间隔。为了最大化间隔的宽度, 我们要最大化什么变量组成的方程?
5. 一旦找到支持向量及其权重, 如何找到分类向量 (即称为 w 的向量)?
6. 为列于表 10-10 中的数据找到一个合适的 SVM 分类器对其分类。

表 10-10 用于 SVM 分类的数据

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	y
1	1	0	0	0	1	0	1	1
1	0	0	0	0	0	0	0	-1
0	0	0	1	1	1	1	0	-1
0	1	0	0	1	0	0	0	-1
1	0	0	0	0	1	0	1	-1
0	1	1	0	1	1	0	1	1
1	1	1	1	0	1	0	1	1
0	1	1	1	0	1	0	0	1
0	0	0	0	0	0	1	1	-1

7. 使用 10.5 节中的代码产生螺旋数据集, 并使用邻近支持向量机找到一个分类器。使用不同的核并在检验数据上比较其在分类误差上的性能。
8. 使用 10.5 节的等式产生棋盘数据集, 并使用邻近支持向量机寻找一个分类器。使用不同的核并在检验数据上比较其在分类误差上的性能。
9. 从 O. L. Mangasarian 的网站 (<http://www.cs.wisc.edu/~olvi/cs525.html>) 下载乳腺癌的数据集, 并使用线性规划寻找一个线性分类器 (本书光盘中也包含该数据集)。

参考文献

- [1] T.M. Mitchell, *Machine Learning*, McGraw-Hill, Boston, 1997.
- [2] V. Vapnik, *Statistical Learning Theory*, Wiley, NY, 1998.
- [3] B.D. Ripley, *Pattern Recognition and Neural Networks*, Cambridge University Press, 1996.
- [4] V. Vapnik, *Theory of Pattern Recognition*, Nauka, Moscow, 1974.

- [5] V. Vapnik and A. Chervonenkis, *Theory of Pattern Recognition* [in Russian], Nauka, Moscow, 1979.
- [6] C.J.C. Burges, A tutorial on support vector machines for pattern recognition, *Data Mining and Knowledge Discovery*, 2(2): pp. 121–167, 1998.
- [7] J. Mercer, Functions of positive and negative type and their connections with the theory of integral equation, *Philos. Trans. Roy. Soc. London*, A209: pp. 415–446, 1909.
- [8] C.M. Bishop, *Neural Networks for Pattern Recognition*, Claredon Press, Oxford, 1995.
- [9] E. Osuna, R. Freund and F. Girosi, *Support Vector Machines: Training and Applications*, AI Memo 1602, MIT, May 1997.
- [10] Robert Burbidge and Bernard Buxton, *An Introduction to Support Vector Machines for Data Mining*, <http://stats.ma.ic.ac.uk/rdb/public-html/pubs/yor12-svm-intro.html>.
- [11] Glen Fung and Olvi Mangasarian, *Proximal Support Vector Machine Classifiers*, proceedings, KDD 2001.
- [12] Olvi Mangasarian, *Data Mining with support vector machines*, *Data Mining Institute Report* 01–05, University of Wisconsin, Madison. May 2001.

第11章 聚类分析

11.1 引言

将对象划分为有意义的组群的能力是智能最重要的模式之一。人类可以轻而易举地完成这项任务。例如，早在孩童时代，人们便学习区分猫和狗、苹果和橙子。但是让计算机来完成这项自动分组工作是一项非常困难的，并且通常是不适定的问题(ill-posed problem^①)。

聚类分析(clustering analysis)是一种探查数据结构的工具。聚类分析的核心是聚类，即将对象划分为簇，使得同一个簇的对象相似，而不同簇的对象相异。对象可以通过某些度量(如属性、特征)或与其他对象的关系(例如，逐对距离、相似性)来描述。与分类不同，聚类不需要以先验标识符来标定数据类别标号的假定。因此，聚类属于非监督学习技术，而分类属于监督学习技术。

对急剧增长的数据加以组织和从数据中学习有价值信息的需要，使得聚类成为一个非常活跃的研究领域。不采用概括技术，人们很难从充斥着大量信息的数据库中发现知识。基本的统计量(如均值、方差)或者直方图可以提供对于数据的初步感觉。然而，聚类分析可以揭示对象之间、特征之间以及对象和特征之间错综复杂的关系。

聚类在很多领域被广泛应用，包括人工智能、生物学、客户关系管理、数据压缩、数据挖掘、信息检索、图像处理、机器学习、市场营销、医药、模式识别、心理学和统计学。例如，在生物学领域，聚类被用来依据物种特征自动建立物种分类。当前，研究人员对根据基因序列数据建立系谱树产生了浓厚的兴趣。聚类的另一个应用是帮助更好地理解细胞内生物学过程中的基因功能。分析基因表达数据的一个关键步骤是发现基因组，使得每个基因组具有相似的表达形式。一个日益发展的应用领域是客户关系管理，在该领域中，可以从多样的接触点(网上冲浪、收银机交易、呼叫中心业务)方便地收集数据，这些数据中包含关于顾客行为的有价值知识，利用这些知识可以优化市场营销方案、制定促销和定价策略。由于数据量巨大而且琐碎，从中抽取知识是非常困难的，有时非常显而易见的东西也会被忽略。聚类在挖掘过程中是至关重要的，它可以通过对具有相似特征的客户进行分组而将数据概括到一个可以管理的级别。

聚类分析是一项基本分析技术，不需要对组数目或组结构做任何假设。分组基于对象间相似性或者距离(相异性)进行。聚类的输入是相似性度量，或者是可以从中计算相似性的数据。

为了说明定义自然分组的困难本质，考虑将一副普通扑克牌中的16张人头牌进行分类。一些简单的分组如图11-1所示。

① 在数学文献中，ill-posed problem 译作“不适定问题”。不适定问题是指没有解，或解不唯一的问题。——译者注

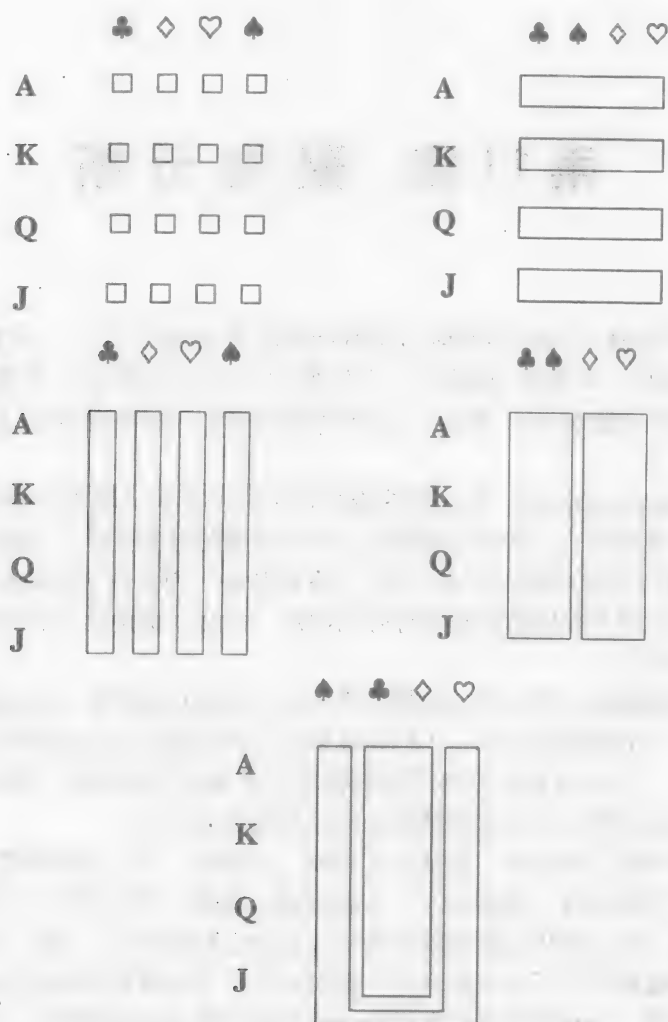
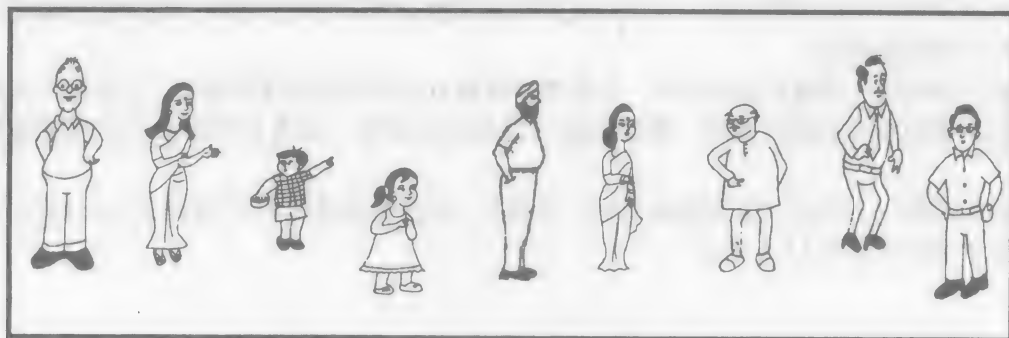


图 11-1 纸牌分组

然而，通过另一个例子(如图 11-2 所示)，可以说明聚类是非常主观的(或者随具体问题而定的)。仔细观察图中所示人物。



a)

图 11-2 聚类是主观的

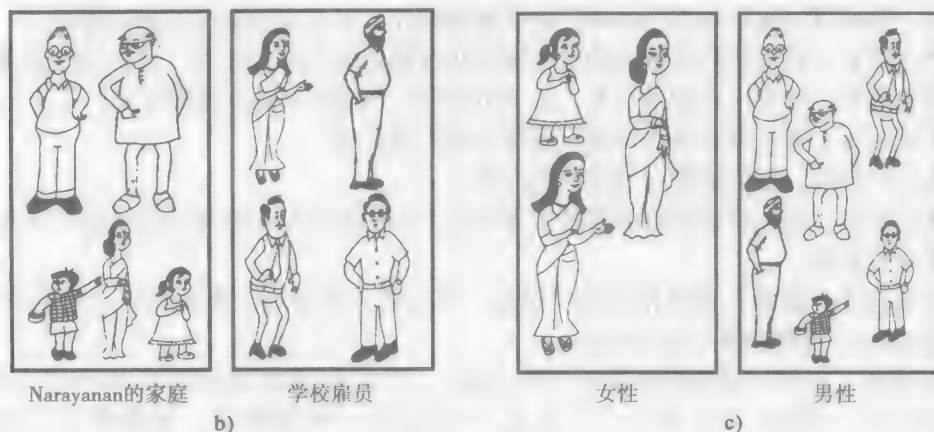


图 11-2 (续)

在大多数实际的聚类分析中, 研究者对如何区分好的分组与坏的分组问题有充分的了解。

聚类分析的首要目的是发现项(或者变量)的自然分组。于是, 首先我们必须拟订一个量化的尺度来度量对象之间的联系。这些尺度主要指相似性度量, 并且主要是表示对象间距离的统计度量。

11.1.1 相似性及其度量

从复杂数据中提取相对简单分组结构的主要工作是找到一个“紧密度”或相似性度量。韦伯斯特字典将相似性定义为相似的性质或状态; 相似; 类似之处; 特征相似。定义相似性是件很困难的事情, 但是“当我们看到它的时候, 我们即可领会”。如图 11-3 所示, 观察两个哺乳动物的相似性。相似性的真正意义是一个哲学问题, 但是, 在数据挖掘中, 我们必须采用实用主义的方法。我们基于特征来测量相似性。



图 11-3 相似性虽然很难定义, 但一目了然

有时,我们可以采用很完美的特征来度量相似性,但大多数情况下我们需要:

- 产生特征。假设我们希望寻找身体健康状况相似的人群,此时,身高和体重便不是有用的特征,我们需要知道这些人的 BMI ($\text{BMI} = \text{体重}(\text{千克}) / \text{身高}^2(\text{米})$)。
- 提炼特征。获得的特征有可能含有噪声或为离群值。
- 规范化特征。我们需要对特征进行变换。
- 减少特征。我们可能获得的特征太多以至于不能进行有效的相似性测量,因此可能需要进行降维。

不存在神奇的黑箱来测量相似性。然而,存在两条有用的一般性技巧:特征投影(feature projection)和编辑距离(edit distance)。

特征投影:当我们将数据投影到特征空间,(以恰当的方式测量的)特征空间中的距离就是相似度。例如,有一群鸟,包含九个不同品种:吸蜜蜂鸟、紫喉蜂鸟、金喉红顶蜂鸟、茶隼、矛隼、秃头鹰、金雕、草原隼等。现在考虑特征①体重,②喙长与身长之比。我们用这些特征将不同种类的鸟投影到特征空间,如图 11-4 所示。在这个空间中,以不同种类鸟之间的距离作为相似性度量。显而易见,蜂鸟和隼在特征空间中被明显地进行了区分。

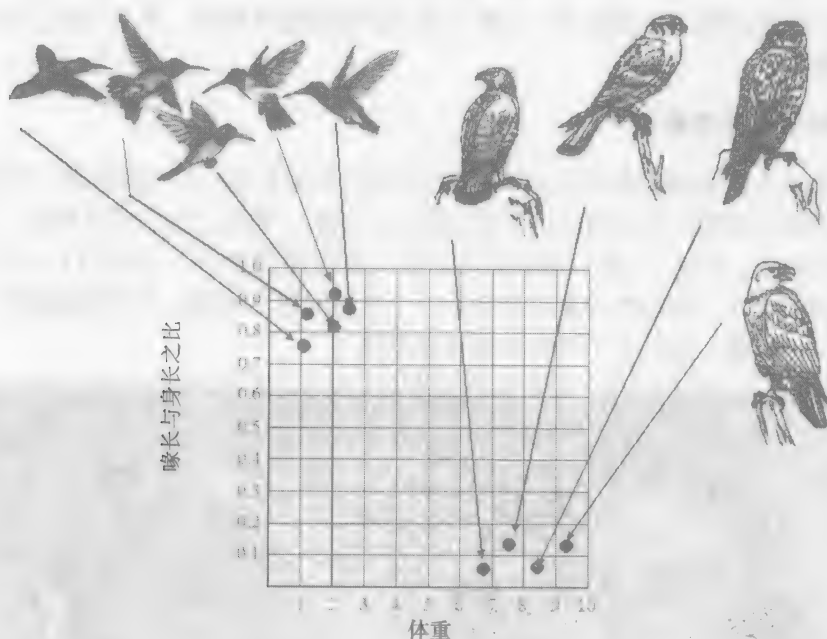


图 11-4 特征投影

另外一个例子是费希尔(Fisher)的鸢尾属植物数据集(见图 11-5)。现有三种植物属于同一植物物种。我们选取花瓣长度和宽度作为特征,并将其投影到特征空间。

我们注意到,现在对象间的相似性依赖于我们测量的特征(和距离度量本身)。图 11-6 说明相对于某些特征来说两个个体之间非常接近,而相对于其他某些特征来说相距很远。

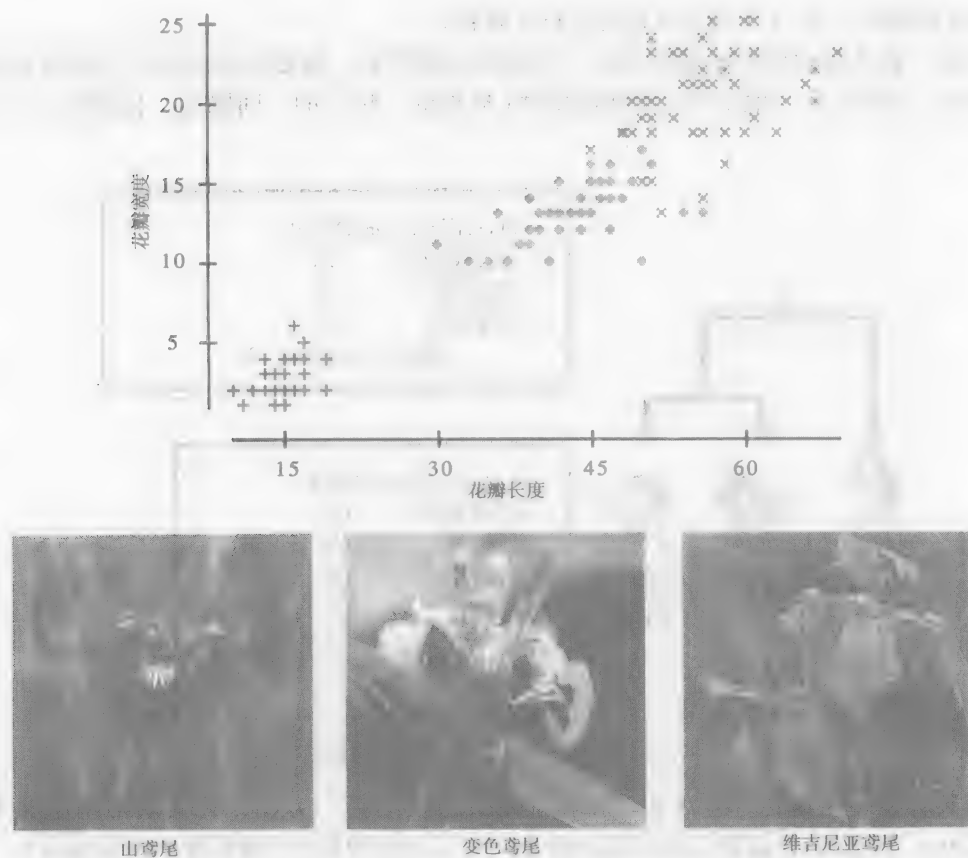


图 11-5 鸢尾属植物物种和基于花瓣长度及宽度投影的特征空间



图 11-6 相似性依赖于特征

编辑距离 (edit distance): 第二种度量两个对象之间相似性的方法是, 将一个对象转换为另一个对象, 并且测量其所花费的工作量。这种工作量的度量, 更确切地说, 转换的“代

价”就是相似性。图 11-7 对这个概念进行了解释。

通常,在选择相似性度量时掺杂了大量的主观因素。重要的考虑因素包括变量的本质(离散的、连续的、二值的)或者测量刻度(标称的、顺序的、间隔的、比值的)以及主题知识。

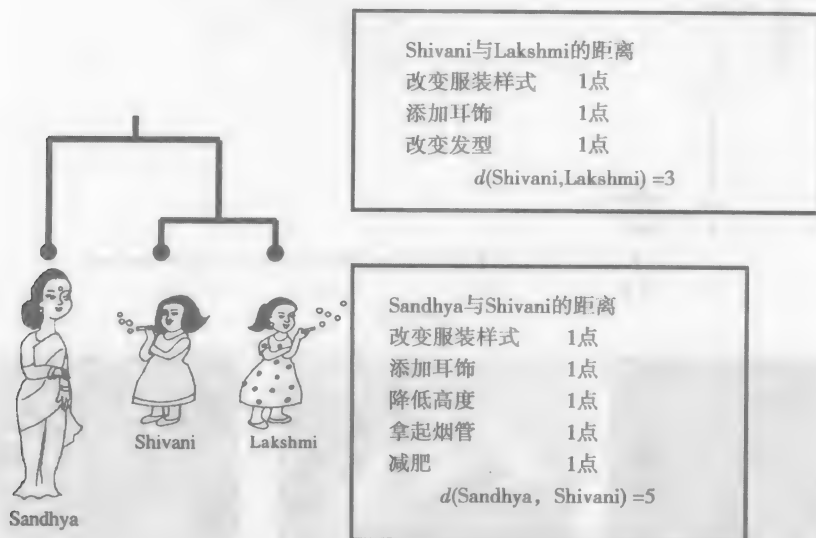


图 11-7 采用编辑距离测量相似性

当所有项被聚类后,通常某种距离表明邻近度。另一方面,变量通常基于相关系数(correlation coefficient)或者关联度量而聚合。为了测量相似性(通常我们测量相异性——距离度量,相似程度越高距离越小),我们测量对象的相关特征(针对当前特定问题的),得到一个相似性/距离的数值度量。图 11-8 解释了这个概念。



图 11-8 聚类内的距离表示接近度

距离度量定义: 令 O_1 和 O_2 表示客观世界中的两个对象, O_1 和 O_2 之间的距离(相异性)是一个实数,用 $\text{distance}(O_1, O_2)$ 或者 $d(O_1, O_2)$ 表示(参见图 11-8)。

连接(joining)或树聚类(tree clustering)方法在形成聚类时使用对象之间的相异度或者距离。这类距离可以是基于单维或者多维的。例如,如果要对方便食品进行聚类,可以考虑它们所包含的卡路里、价格、主观的嗜好排名等。最直接的计算多维空间中对象间距离的方法就是计算欧氏距离。在二维或三维空间中,这种度量就是该空间中对象间的几何距离(如同采用尺子进行测量)。对于树聚类方法来讲,它并不在意距离是否是现实世界中的真实距离,还是其他对研究者来说更具意义的推导出的距离度量,这主要取决于研究者根据特定应用选择合适的方法。

欧氏距离(Euclidean distance):这是一种最常用的距离。它可以简单表示为多维空间中的几何距离,计算公式为:

$$Distance(O_i, O_j) = 2 \sqrt{\sum_{k=1}^n (O_{ik} - O_{jk})^2} \quad (11.1)$$

例如,考虑表 11-1 所列的数据。

表 11-1 相关数据

对象/变量	X_1	X_2	X_3	X_4
Q_1	5	6	4	9
Q_2	8	9	3	2
Q_3	3	4	5	3

按特定顺序排列的对象的特征值,可以把对象看作是多维空间中的点。在这个例子中,由于具有四个变量,因此空间维数为 4。对象间的距离为:

$$Distance(O_1, O_2) = \sqrt{(5-8)^2 + (6-9)^2 + (4-3)^2 + (9-2)^2} = 8.25$$

$$Distance(O_1, O_3) = \sqrt{(5-3)^2 + (6-4)^2 + (4-5)^2 + (9-3)^2} = 6.7$$

$$Distance(O_2, O_3) = \sqrt{(8-3)^2 + (9-4)^2 + (3-5)^2 + (2-3)^2} = 7.42$$

注意,欧氏距离(以及平方距离)通常是由原始数据计算得到,而非规则化后的数据。该方法具有某些优势(例如,不增加新对象不影响任意两个对象之间的距离,即使新增对象是离群点)。然而,不同维间的尺度选择对聚类结果有着显著的影响。假如,其中一维表示测量的长度,用厘米表示,如果将它换算为毫米表示(乘以 10),那么欧氏距离或者平方欧氏距离(从多维计算得到)的计算结果将受很大影响,聚类分析结果也会因此而发生变化。

街区距离(city-block distance):街区距离(又称曼哈顿距离)仅仅是维上的平均差。在通常情况下,这种距离度量产生的结果类似于欧氏距离。要注意的是,在该距离度量中,离群点的作用被消弱(由于没有取平方)。街区距离的计算公式为:

$$Distance(O_i, O_j) = \frac{1}{n} \sum_{k=1}^n |O_{ik} - O_{jk}| \quad (11.2)$$

用表 11-1 中的数据计算街区距离如下:

$$Distance(O_1, O_2) = \frac{1}{4} (|5-8| + |6-9| + |4-3| + |9-2|) = 3.5$$

$$Distance(O_1, O_3) = \frac{1}{4} (|5-3| + |6-4| + |4-5| + |9-3|) = 2.75$$

$$Distance(O_2, O_3) = \frac{1}{4} (|8-3| + |9-4| + |3-5| + |2-3|) = 3.25$$

切比雪夫距离(Chebychev distance):在某些情况下,当需要利用任何一维变量来区分两

个对象时, 可以选择采用切比雪夫距离。该距离度量的计算公式为:

$$Distance(O_j, O_i) = \text{Max}_k (|O_{ik} - O_{jk}|) \quad (11.3)$$

$$Distance(O_1, O_2) = \text{Max}(|5-8|, |6-9|, |4-3|, |9-2|) = 7$$

$$Distance(O_1, O_3) = \text{Max}(|5-3|, |6-4|, |4-5|, |9-3|) = 6$$

$$Distance(O_2, O_3) = \text{Max}(|8-3|, |9-4|, |3-5|, |2-3|) = 5$$

幂距离(power distance): 当人们希望增大或减小那些对象间差异最大变量的渐进权重系数时, 可以采用幂距离。幂距离的计算公式为:

$$Distance(O_i, O_j) = \left(\sum_{k=1}^n |O_{ik} - O_{jk}|^p \right)^{\frac{1}{p}} \quad (11.4)$$

其中, r 和 p 为用户自定义参数。一些计算例子可以说明该距离度量的行为。参数 p 用来控制每个维差值的渐进权重(progressive weight); 参数 r 控制对象间较大差值的渐进权重。如果将 r 和 p 同时设为 2, 则幂距离等同于欧氏距离。

差异百分率(percent disagreement): 如果分析中的数据各维在本质上是明确分来的, 这种度量尤其适用。距离计算公式为:

$$Distance(O_i, O_j) = \frac{100 \times [\text{Number of } (O_{ik} \neq O_{jk})]}{n} \quad (11.5)$$

例如, 考虑表 11-2。

病人 1 和病人 2 仅仅在年龄组的值上不同, 所以其距离为:

$$Distance(O_1, O_2) = \frac{100 \times 1}{4} = 25\%$$

病人 1 和病人 3 在两个属性上存在不同, 因此

$$Distance(O_1, O_3) = \frac{100 \times 2}{4} = 50\%$$

同理, 病人 2 和病人 3 的距离为:

$$Distance(O_2, O_3) = \frac{100 \times 3}{4} = 75\%$$

表 11-2 具有分类属性的数据

病人编号	性别	年龄组	收入水平	BP
1	M	20-30	低	正常
2	M	30-40	低	正常
3	F	20-30	中	正常

二元属性对象的相似性: 当项不能用有意义的 p 维测量表示时, 项对之间的比较通常根据某些特征的存在和缺失完成的。相似的项通常具有更多的共同特征。可以通过引入一个二元变量来描述是否具有某种特征, 如果具有则该特征变量值为 1, 否则变量值为零。考虑如下例子。表 11-3 列出了 5 个个体的特征。

表 11-3 5 个个体的特征

	身高	体重	嘴唇	头发	用手习惯	性别
个体 1	155cm	65kg	薄	卷曲	右手	女
个体 2	172cm	75kg	厚	直发	右手	男
个体 3	162cm	68kg	薄	卷曲	右手	男
个体 4	168cm	62kg	厚	卷曲	右手	女
个体 5	175cm	80kg	厚	卷曲	左手	男

定义6个变量 X_1 、 X_2 、 X_3 、 X_4 、 X_5 、 X_6 ，其取值分别为：

$X_1 = 1$ 如果身高 $\geq 165\text{cm}$ ，否则 $X_1 = 0$

$X_2 = 1$ 如果体重 $\geq 70\text{kg}$ ，否则 $X_2 = 0$

$X_3 = 1$ 如果嘴唇厚，否则 $X_3 = 0$

$X_4 = 1$ 如果头发卷曲，否则 $X_4 = 0$

$X_5 = 1$ 如果习惯用右手，否则 $X_5 = 0$

$X_6 = 1$ 如果是男性，否则 $X_6 = 0$

个体1和个体2的6个二元变量的得分如表11-4所示。

表11-4 个体1和个体2的变量得分

	X_1	X_2	X_3	X_4	X_5	X_6
个体1	0	0	0	1	1	1
个体2	1	1	1	0	1	0

现在，我们创建另外一个矩阵(见表11-5)来统计匹配和非匹配的数量。矩阵元素 a 、 b 、 c 、 d 通过如下计算方式获得：

两个个体得分都是“1”的属性数 = $a = 1$ 。

个体1得分为“1”而个体2的得分为“0”的属性数 = $b = 2$ 。

个体1得分为“0”而个体2的得分为“1”的属性数 = $c = 3$ 。

两个个体的得分都为“0”的属性数 = $d = 1$ 。

表11-5 得分矩阵

	个体2		合计
	1	0	
个体1	1 $a = 1$	2 $b = 2$	3
	3 $c = 3$	1 $d = 1$	4
合计	4	3	7

现在，我们如何测量相似性(或者相异性)? 尽管基于前面的计算公式所得的距离可以用来测量相似性，但是其缺陷在于赋予0-0和1-1匹配相同的权重。在某些例子中，一个1-1匹配关于相似性的暗示比0-0匹配更强烈。例如，当把人进行分类时，如果两个人都能阅读梵文，与不具备这个能力的人相比，这是两个人相似的一个强有力的证据。因此，将0-0匹配作用打折扣甚至于完全忽略也是合理的。为了便于差别对待0-0匹配和1-1匹配，人们提出了许多定义相似性系数的方案。表11-6列出了几种相似性度量及其对应的原理。

表11-6 相似性系数

编号	系数	原理
1	$\frac{a+d}{p}$	给0-0匹配和1-1匹配相等的权重
2	$\frac{2(a+d)}{2(a+d)+b+c}$	给0-0匹配和1-1匹配两倍的权重
3	$\frac{a+d}{a+d+2(b+c)}$	给不匹配对两倍权重
4	$\frac{a}{p}$	分子中无0-0匹配

(续)

编 号	系 数	原 理
5	$\frac{a}{a+b+c}$	将 0-0 匹配视为不相关
6	$\frac{2a}{2a+b+c}$	不给 0-0 匹配分配权重, 给 1-1 匹配双倍分配两 倍权重
7	$\frac{a}{a+2(b+c)}$	分子、分母不包含 0-0 匹配, 不匹配对分配两 倍权重
8	$\frac{a}{b+c}$	不考虑匹配与非匹配之比以及 0-0 匹配

采用相似性系数 1 给所有匹配分配相同的权重, 我们有

$$\frac{a+d}{p} = \frac{1+0}{6} = \frac{1}{6}$$

继续采用表 11-6 中的相似性系数 1, 我们计算表 11-3 中其余个体对的相似度。计算结果用 5×5 对称矩阵表示(见图 11-9)。

	个 体				
	1	2	3	4	5
1	1				
2	1/6	1			
个体 3	4/6	3/6	1		
4	4/6	3/6	2/6	1	
5	0	5/6	2/6	2/6	1

图 11-9 相似性矩阵

基于相似度矩阵的值, 我们可以得出结论: 个体 2 和个体 5 最相似, 而个体 1 和个体 5 相似性最小。其余的个体对的相似度位于这两个极值中间。

我们已经讲述了如何构造距离和相似性, 并且总是可以由距离来构造相似性。例如, 我们可以设

$$S_{ik} = \frac{1}{(1 + d_{ik})} \quad (11.6)$$

其中, $1 \geq S_{ik} \geq 0$ 表示项 i 和 k 之间的相似性, d_{ik} 为对应的距离。由于满足非负条件并且最大相似度 $S_{ii} = 1$, 度量

$$d_{ik} = \sqrt{2(1 - S_{ik})} \quad (11.7)$$

具有距离的特性。

11.1.2 聚类的基本类型

广义来讲, 聚类可以分为两类(参见图 11-10):

- 1) 划分算法: 在划分算法中, 我们构造不同的划分并用某种标准来评价它们。
- 2) 层次算法: 在层次算法中, 我们用某种标准来对一组对象进行层次分解。

图 11-10 显示了用两种方法所得到的不同结果。

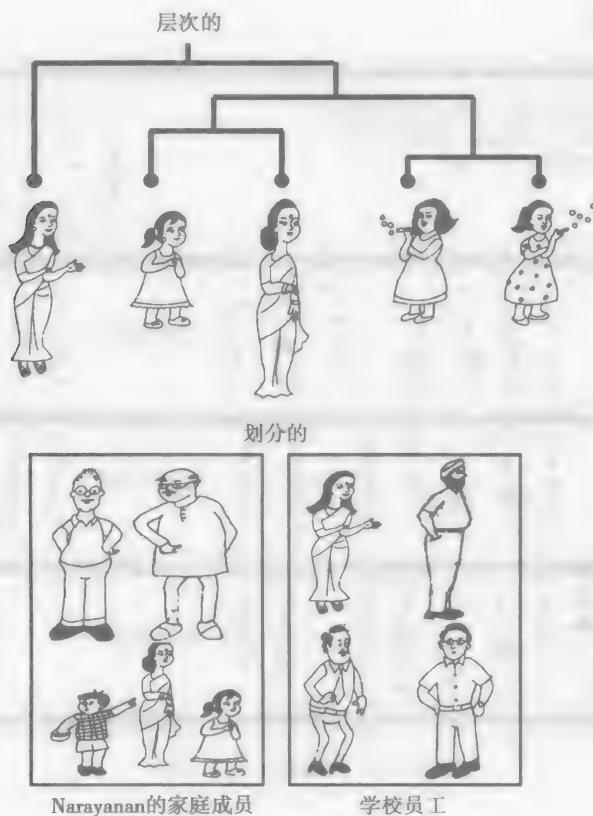


图 11-10 基本聚类形式

聚类算法的期望特征：下面列出了聚类算法的期望特征：

- 可伸缩性(时间和空间)
- 可以处理不同类型数据
- 可以发现任意形状的簇
- 确定输入参数所需要的领域知识需求最少
- 能够应对噪声和离群点
- 对输入记录顺序不敏感
- 可以整合用户特定的约束
- 可解释性和可用性

层次聚类：我们可以通过两种方法构造层次树：

自底向上(凝聚)：最初，我们假定所有项属于一个单独的簇，然后寻找最佳配对并合并成一个新的簇。在图 11-11 中，聚类过程由底部开始，最终结果显示在最上面。

自顶向下(分裂)：开始将所有数据看作一个簇，考虑所有可能的方法，将簇一分为二。选择最佳划分，并递归地在这两个簇上继续进行划分。

最常用的一个算法是凝聚方法，下面我们将对其进行详细讲述。

凝聚层次聚类：依靠共同的距离度量，聚类过程从寻找距离最近的簇开始，并把这两个簇合并为一个簇。在过程开始之初，让每个对象自成一簇，每个簇都以选定的距离度量定义。然而，当几个对象被连接在一起时，我们如何确定这些新簇之间的距离？图 11-12 和图

11-13 说明了这个问题。

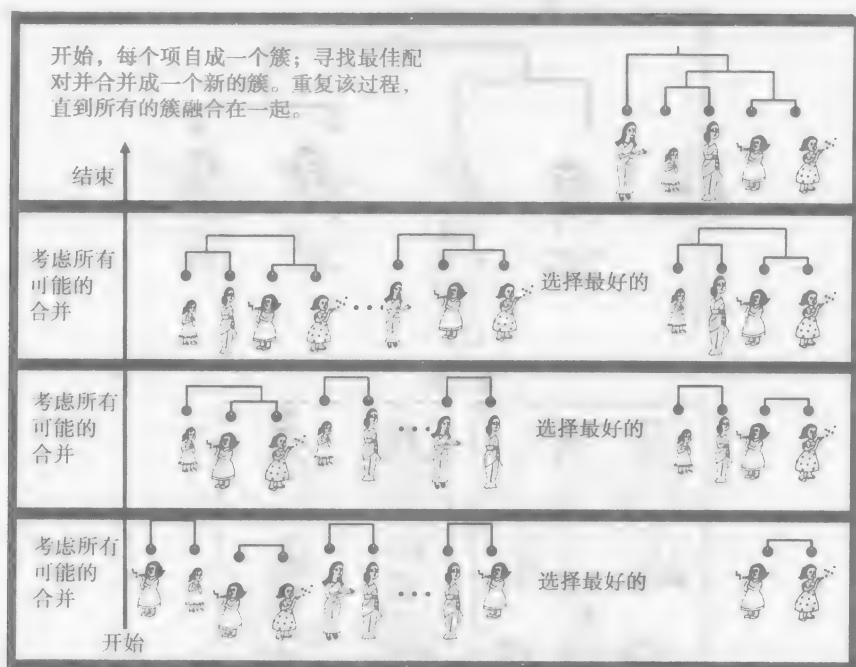


图 11-11 自底向上凝聚聚类

我们由数据库中每对对象间距离的距离矩阵开始

$$d(\text{对象1}, \text{对象2}) = 8$$

$$d(\text{对象4}, \text{对象5}) = 1$$

0	8	8	7	7
	0	2	4	4
		0	3	3
			0	1
				0

图 11-12 层次聚类的第 1 步：生成两两距离矩阵

对于图 11-12，对象 4 和对象 5 间的距离是 1，因此我们合并它们。现在我们只剩四个对象——三个单独的对象和一个包含两个单独对象的合成对象。我们继续生成另一个 4×4 的距离矩阵。由于前三个对象没有发生改变，因此矩阵元素值不变（参照图 11-13）。因为第四个对象为合成对象，所以计算出它与其他三个对象的距离是个问题。我们该如何定义两个合成对象间的距离？

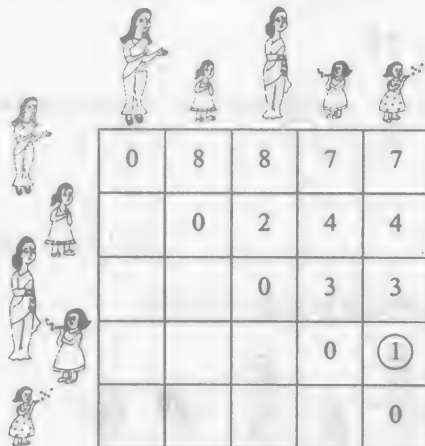
换言之，我们需要一个连接或合并规则来确定何时两个簇已足够相近，可以连接在一起。存在很多种可能性。例如，当两簇中的任意两个对象比各自的连接距离更接近，我们便

可以将两个簇连接在一起。换句话说,我们用簇间的“最近邻”来确定簇间距离。这种方法叫做单连接(single linkage)。这个规则生成“纤维状”类型的簇,即簇间仅仅是靠单个紧密靠近的对象而连接在一起。换一种做法,我们也可以用簇间距离最近的近邻,该方法称为完全连接(complete linkage)。如上所述,还存在其他众多的连接规则。

在凝聚聚类的第一次迭代中,

我们合并了 。因此我

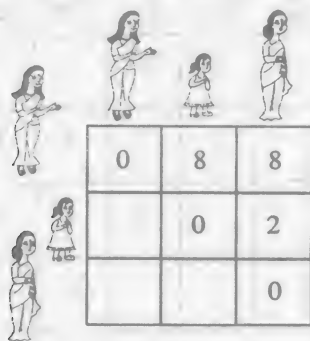
们需要将她们从矩阵中移出。



我们需要增加新的单独聚

类  到我们新的

更小矩阵。



$$d(\text{cluster of girls}, \text{woman 1}) = ?$$

$$d(\text{cluster of girls}, \text{woman 2}) = ?$$

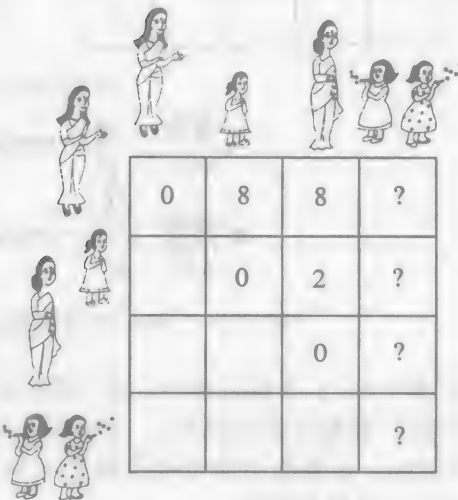


图 11-13 层次聚类中连接(凝聚)的需求

单连接(最近邻): 如上所述,在该方法中,两个簇的距离由不同簇的两个最靠近的对

象间的距离决定。简言之,簇的距离是属于不同簇的两个样本间的最近距离(参见图 11-14 和图 11-15)。在某种意义上,这个规则将对象串起来形成簇,聚类结果趋向于表示长“链”。

$$d(c_1, c_2) = \min_{o \in C_1, O \in C_2} \{d(o, O)\} \quad (11.8)$$

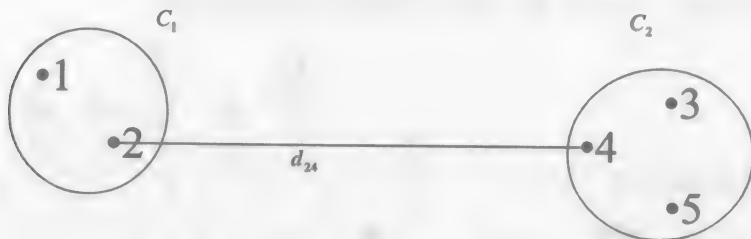
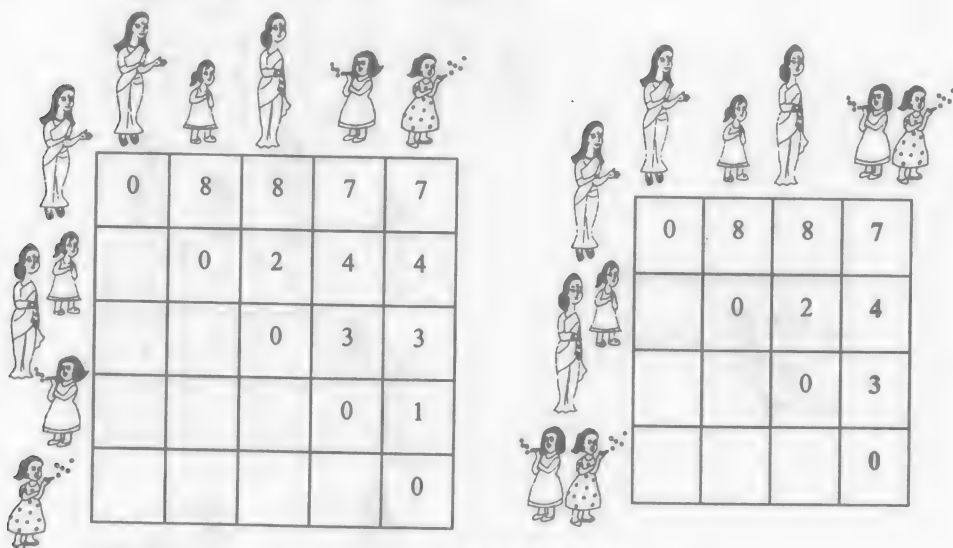


图 11-14 单连接

图 11-15 用图形象地解释了该方法。



采用单连接(最近邻)

$$d(\text{Group 1}, \text{Group 2}) = \min[d(\text{Person 1}, \text{Person 2}), d(\text{Person 3}, \text{Person 4})] = 4$$

$$d(\text{Group 1}, \text{Group 3}) = \min[d(\text{Person 1}, \text{Person 5}), d(\text{Person 3}, \text{Person 6})] = 7$$

图 11-15 采用单连接规则进行层次聚类例子

完全连接(最远邻):在该方法中,两个簇的距离由隶属于不同簇的距离最远的两个对象间的距离所决定(即最远邻的距离)。当对象在事实上自然形成截然不同的簇时,这种方法的效果通常很好。如果簇倾向于有点修长或者具有“链状”特性时,不适用该方法。图 11-16说明了完全连接聚类计算。

$$d(c_1, c_2) = \min_{o \in C_1, O \in C_2} \{d(o, O)\} \quad (11.9)$$

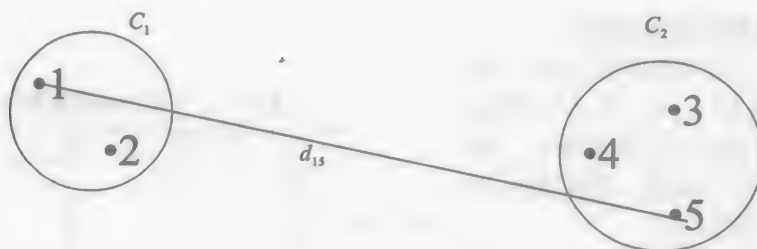


图 11-16 完全连接

组平均 (unweighted pair-group average): 在该方法中, 两个簇的距离就是属于不同簇的所有对象对的距离的平均值 (参见图 11-17)。当对象形成自然的不同簇时, 该方法非常有效。然而, 该方法对修长、“链状”簇的效果也很好。在 Sneath 和 Sokal 的书中[1], 用缩写 UPGMA 来表示该方法。

$$d(c_1, c_2) = \frac{1}{n_1 n_2} \sum_{o \in C_1, O \in C_2} \{d(o, O)\} \quad (11.10)$$

其中, n_1 和 n_2 分别为两个簇包含的样本数。

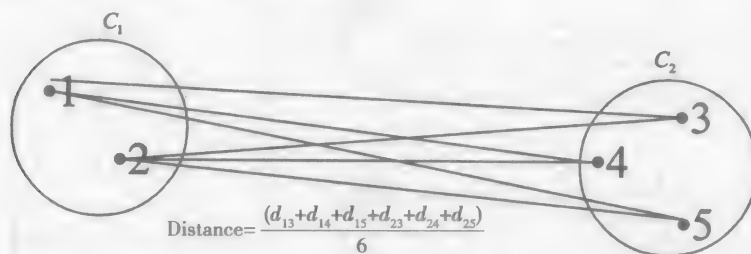


图 11-17 逐对平均距离

加权组平均 (weighted pair-group average): 除了在计算时各个簇的大小 (簇所包含对象数) 被用作权重外, 该方法与组平均方法一样。当簇的大小可能存在严重不对等的情形时, 应该采用该方法 (而不是前面方法)。加权的组平均距离可以采用缩写 WPGMA 表示。

组质心 (unweighted pair-group centroid): 簇的质心 (centroid) 就是簇在多维空间的平均点。从某种意义上说, 它是簇的重心 (centre of gravity)。在该方法中, 两个簇的距离由簇的质心之间的距离确定。Sneath 和 Sokal (1973) 用缩写 UPGMC 表示该方法。

加权组质心 (weighted pair-group centroid): 除了计算时考虑簇的大小 (簇所包含对象数) 差异外, 该方法与上面的组质心方法一样。因此, 当考虑簇大小的差异时, 该方法优于不加权的组质心方法。Sneath 和 Sokal (1973) 用缩写 WPGMC 表示该方法。

沃德法 (Ward's method): 该方法不同于其他所有方法, 它采用方差分析法来计算簇间距离。简言之, 该方法试图最小化每步形成的任意两个簇的平方和 (SS)。关于该方法的详细描述, 请参阅 Ward (1963) [2]。通常, 该方法被认为非常有效, 但是它产生的簇一般比较小。

采用不同连接的聚类的例子

考虑表 11-7 给出的距离阵列。表中, 值表示对象 O_1, O_2, O_3, O_4 间的距离。为了说明连接及其各种形式, 通过表 11-7 中给出的数据, 观察相对于上述问题它的各种形式。

表 11-7 4 个对象间相互距离

	O_1	O_2	O_3	O_4
O_1	0			
O_2	1	0		
O_3	11	2	0	
O_4	5	3	4	0

$D = \{d_{jk}\} =$

采用单连接的例子。在距离矩阵中, 选择最小的距离值, 这里 $d(O_1, O_2)$ 是最小的且等于 1。采用单连接度量作为距离度量, 合并 O_1 和 O_2 形成新的对象 $O_{1,2}$ 。

表 11-8 显示了有关的各个步骤。现在我们有三个对象, 其中对象 $O_{1,2}$ 是一个子簇(合成对象)。为进一步聚类, 计算这三个对象相互间的距离。

表 11-8 应用单连接规则的距离矩阵

	O_1	O_2	O_3	O_4
O_1	0			
O_2	1	0		
O_3	11	2	0	
O_4	5	3	4	0
	$O_{1,2}$	O_3	O_4	
$O_{1,2}$	0			
O_3	2	0		
O_4	3	4	0	
	$O_{1,2,3}$	O_4		
$O_{1,2,3}$	0			
O_4	3	0		

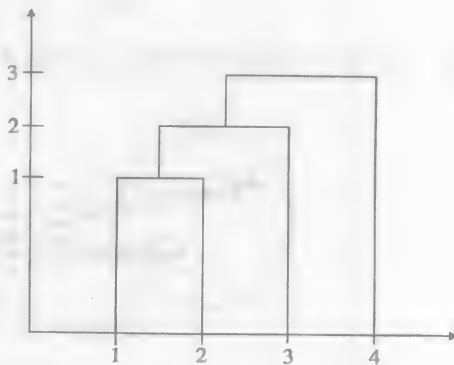


图 11-18 单连接

依据单连接, 可知

$$\begin{aligned} \text{Distance}(O_{1,2}, O_3) &= \min\{\text{Distance}(O_1, O_3), \text{Distance}(O_2, O_3)\} \\ &= \min(11, 2) = 2 \end{aligned}$$

上面所有距离来源于表 11-7。类似地,

$$\begin{aligned} \text{Distance}(O_{1,2}, O_4) &= \min\{\text{Distance}(O_1, O_4), \text{Distance}(O_2, O_4)\} \\ &= \min(5, 3) = 3 \end{aligned}$$

按照表 11-7, $\text{Distance}(O_3, O_4)$ 等于 4。

有了上面三个距离, 我们构造第二个矩阵如表 11-9 所示。它与表 11-8 中的第二个矩阵相同。

表 11-9 采用单连接的簇间距离

	$O_{1,2}$	O_3	O_4
$O_{1,2}$	0		
O_3	2	0	
O_4	3	4	0

得到簇间距离的单连接规则可写为:

$$Distance(P+Q, R) = \min\{Distance(P, R), Distance(Q, R)\} \quad (11.11)$$

其中, $P+Q$ 表示两个子簇的并。

根据单连接准则, 由于对象 $O_{1,2}$ 和 O_3 是最接近的对象, 因此现在将对象 $O_{1,2}$ 和 O_3 结合生成新的簇 $O_{1,2,3}$ 。

表 11-8 中的最后一个矩阵的元素是通过如下公式得到的:

$$\begin{aligned} Distance(O_{1,2,3}, O_4) &= \min\{Distance(O_{1,2}, O_4), Distance(O_3, O_4)\} \\ &= \min(3, 4) = 3 \end{aligned}$$

注意, 计算这个距离时, 我们只需要参考上面步骤得到的矩阵, 即表 11-9 中给出的矩阵。

最后, 得到合成的簇 $O_{1,2,3,4}$, 形成的聚类的层次可以用一个树状图表示, 如图 11-18 所示。

根据图 11-18 所示的树状图, 采用 1 个距离单位, 对象 O_1 和 O_2 构成一个簇; 采用 2 个距离单位, 对象 O_1 、 O_2 和 O_3 组成一个簇; 当采用 3 个距离单位时, 对象 O_1 、 O_2 、 O_3 和 O_4 构成一个复合簇。

采用完全连接的例子。为了解释完全连接, 考虑上面的例子。其过程和单连接类似, 只是规则变为选择 R 的对象和簇 P 、 Q 的对象间的最大距离, 使得假设所有成员都在该最大距离之内, 即

$$Distance(P+Q, R) = \max\{Distance(P, R), Distance(Q, R)\}$$

让我们回到矩阵的例子。在这个例子中, 对象 O_1 和 O_2 最接近, 合并成第一个子簇 $O_{1,2}$ 。现在我们采用完全连接规则构造簇间距离矩阵。

$$\begin{aligned} Distance(O_{1,2}, O_3) &= \max\{Distance(O_1, O_3), Distance(O_2, O_3)\} \\ &= \max(11, 2) = 11 \end{aligned}$$

类似地,

$$\begin{aligned} Distance(O_{1,2}, O_4) &= \max\{Distance(O_1, O_4), Distance(O_2, O_4)\} \\ &= \max(5, 3) = 5 \end{aligned}$$

按照表 11-7, $Distance(O_3, O_4)$ 等于 4。

有了上述三个距离, 我们构造表 11-10 所示的矩阵。

表 11-12 列出了第三个矩阵, 它显示了整个过程。

表 11-10 中矩阵的最小值为 4, 因此我们将对象 O_3 和 O_4 合并生成新的对象簇 $O_{3,4}$ 。

我们继续寻找下一个簇间距离矩阵。我们现在只剩两个簇 $O_{1,2}$ 和 $O_{3,4}$ (见表 11-11)。

表 11-10 采用完全连接的簇间距离

	$O_{1,2}$	O_3	O_4
$O_{1,2}$	0		
O_3	11	0	
O_4	5	4	0

表 11-11 对象 $O_{1,2}$ 和 $O_{3,4}$ 间距离矩阵

$O_{1,2}$	$O_{3,4}$
$O_{1,2}$	0
$O_{3,4}$	11

为了填充矩阵元素, 我们参照表 11-10 并应用完全连接规则。

$$\begin{aligned} Distance(O_{1,2}, O_{3,4}) &= \max\{Distance(O_{1,2}, O_3), Distance(O_{1,2}, O_4)\} \\ &= \max(11, 5) = 11 \end{aligned}$$

表 11-12 完全连接规则

	O_1	O_2	O_3	O_4
O_1	0			
O_2	1	0		
O_3	11	2	0	
O_4	5	3	4	0

	O_1	O_2	O_3	O_4
O_1	0			
O_2	1	0		
O_3	11	2	0	
O_4	5	3	4	0

	$O_{1,2}$	O_3	O_4	0
$O_{1,2}$	0			
O_3	2	0		
O_4	3	4	0	

	$O_{1,2}$	$O_{3,4}$
$O_{1,2}$	0	
$O_{3,4}$	11	0

树状图如图 11-19 所示。

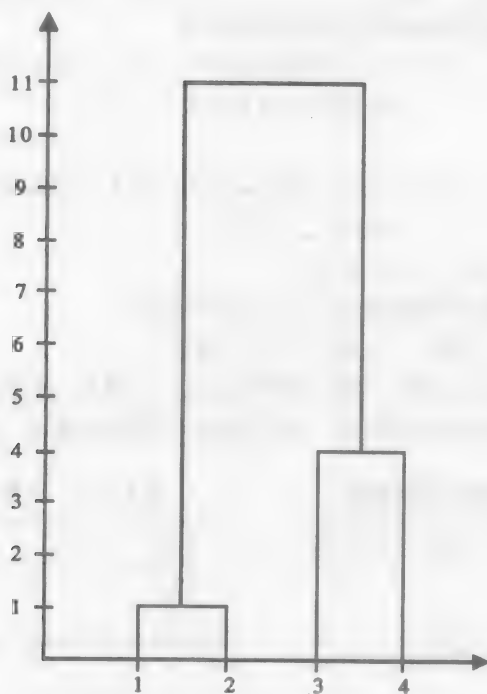


图 11-19 采用完全连接规则表示表 11-7 中数据的树状图

采用平均距离连接的例子。其聚类过程和上面的例子相似，只是连接采用簇 P 、 Q 和对象 R 间的平均距离。

我们先将对象 O_1 和 O_2 结合形成簇 $O_{1,2}$ 。依据平均距离连接规则, 簇间距离矩阵元素计算如下:

$$\begin{aligned} \text{Distance}(O_{1,2}, O_3) &= \left(\frac{1}{2}\right) \{ \text{Distance}(O_1, O_3) + \text{Distance}(O_2, O_3) \} \\ &= \left(\frac{1}{2}\right) (11 + 2) = 6.5 \end{aligned}$$

类似地,

$$\begin{aligned} \text{Distance}(O_{1,2}, O_4) &= \left(\frac{1}{2}\right) \{ \text{Distance}(O_1, O_4) + \text{Distance}(O_2, O_4) \} \\ &= \left(\frac{1}{2}\right) \{ 5 + 3 \} = 4 \end{aligned}$$

由表 11-7 得到 $\text{Distance}(O_3, O_4) = 4$ 。

表 11-13 给出了由此得到的结果矩阵。

参阅表 11-13, 我们观察到矩阵中有两个元素具有最小值。我们可以选择其中的任意一个, 然而这会导致树状图在形状上产生两处轻微改变。我们选择合并对象 O_3 和 O_4 生成 $O_{3,4}$ 。

我们现在构造簇 $O_{1,2}$ 和 $O_{3,4}$ 的簇间距离。

$$\begin{aligned} \text{Distance}(O_{1,2}, O_{3,4}) &= \left(\frac{1}{2}\right) \{ \text{Distance}(O_{1,2}, O_3) + \text{Distance}(O_{1,2}, O_4) \} \\ &= \left(\frac{1}{2}\right) (6.5 + 4) = 5.25 \end{aligned}$$

这个距离也可以按下式计算:

$$\begin{aligned} &\left(\frac{1}{4}\right) \{ \text{Distance}(O_1, O_3) + \text{Distance}(O_1, O_4) \\ &+ \text{Distance}(O_2, O_3) + \text{Distance}(O_2, O_4) \} = \left(\frac{1}{4}\right) (11 + 5 + 2 + 3) = 5.25 \end{aligned}$$

表 11-14 显示了整个处理过程, 对应的树状图如图 11-20 所示。

表 11-14 平均距离连接示意图

	O_1	O_2	O_3	O_4
O_1	0			
O_2	1	0		
O_3	11	2	0	
O_4	5	3	4	0

	O_1	O_2	O_3	O_4
O_1	0			
O_2	1	0		
O_3	11	2	0	
O_4	5	3	4	0



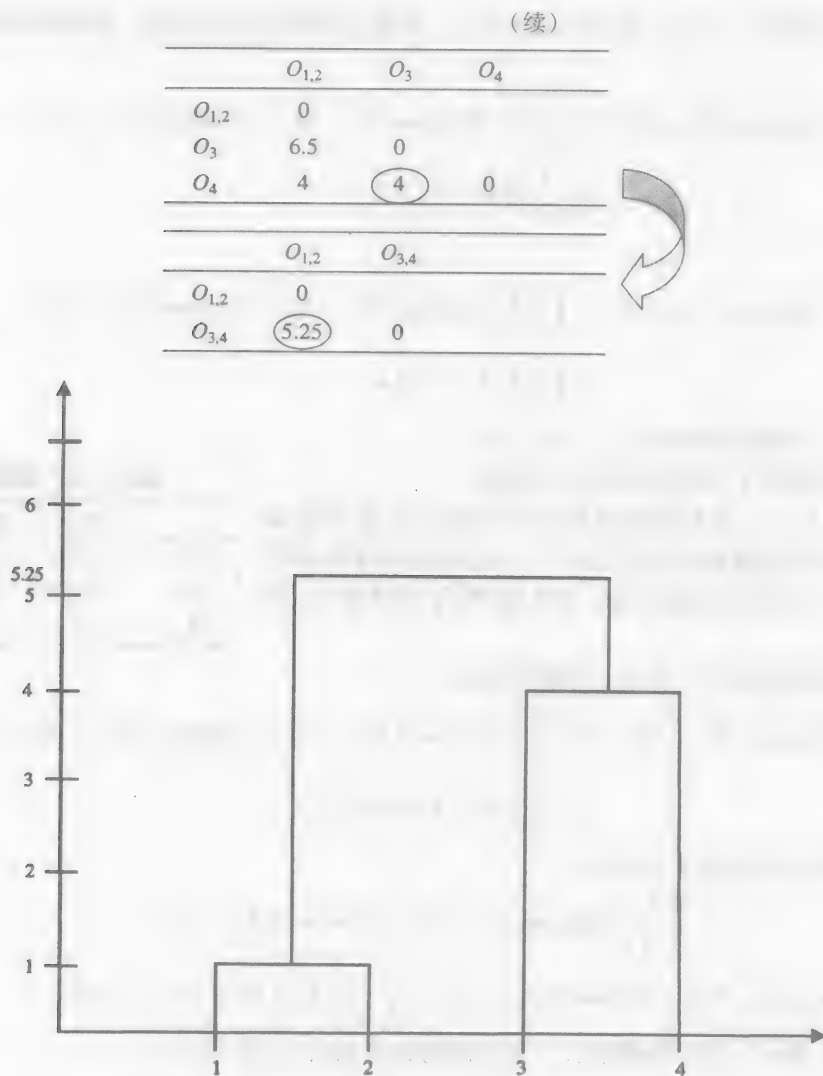


图 11-20 采用平均距离连接规则的表 11-7 数据的树状图

层次聚类的优点

1) 我们可以通过观察树状图来确定正确的簇数目。参见下面的图 11-21。在这个例子中, 两个高度分离的子树强烈暗示是两个簇。(令人遗憾的是, 很少有事情能如此清晰地分开。)

2) 层次的本质很好地反映了人类对某些领域的直觉。

3) 树状图的一个潜在应用是可以用来检测离群点。

层次聚类的缺点: 层次聚类有时会表现出无意义的或者不合逻辑的模式。例如, 在图 11-22 所展示的聚类, 紧密地聚合澳大利亚、安圭拉岛、圣赫勒拿岛等是有意义的, 因为这些国家都曾经是英国的殖民地。但是, 对尼日尔和印度进行紧密聚合是完全不合逻辑的, 它们之间毫无联系。

尼日尔国旗(参见图 11-22)自上而下由橙、白、绿三个平行相等的长方形组成, 白色部分中间有一个橙色圆轮, 象征太阳。橙色象征尼日尔北部边界的沙漠。绿色代表南部和西部美丽富饶的平原和维系它们的尼日尔河, 也象征博爱和希望。白色象征纯洁和希望。

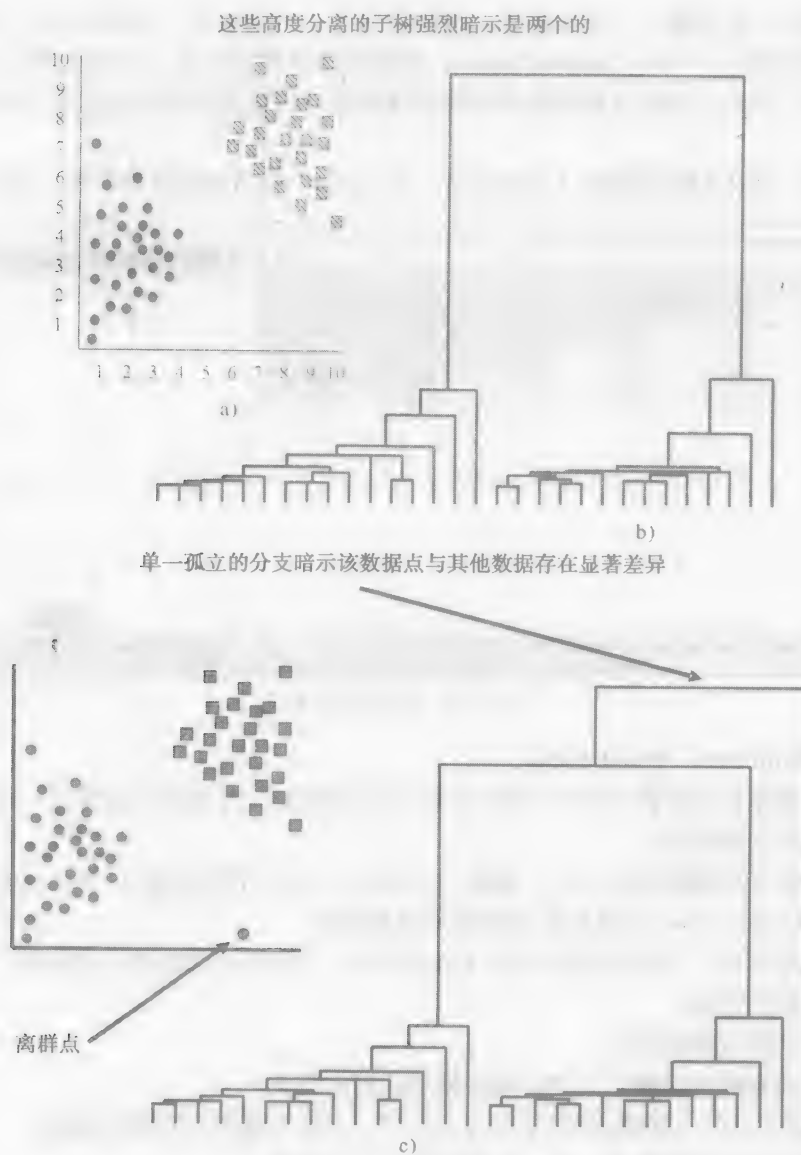


图 11-21 使用树状图发现离群点

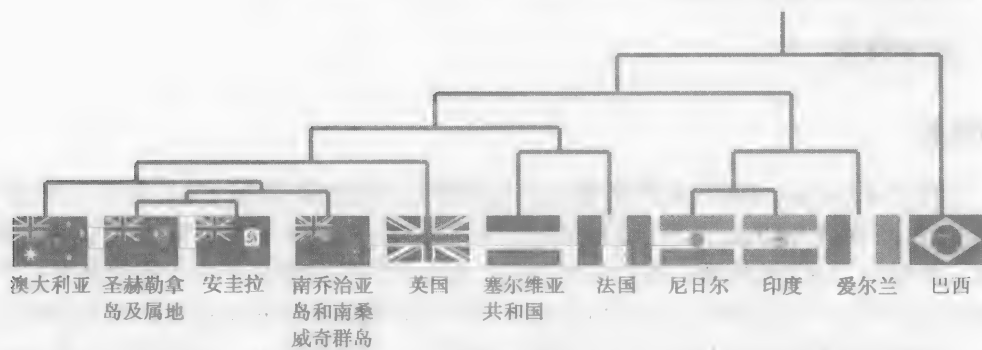


图 11-22 不合逻辑的聚类模式

印度国旗是三色条旗，三部分相等，深橙色在上，白色居中，深绿色在下。在白条中央有一个海军蓝色圆轮，代表 Dharma Chakra，即阿苏迦狮都的法轮。中间的符号“纺车”可以追溯到公元前 2 世纪。橙色象征勇敢和自我牺牲精神；白色象征纯洁和真理；绿色表示生长和吉祥。

演示程序：随书所附光盘第 11 章内包含一个 applet。你可以运行该程序。指令如下所示：



图 11-23 树状图 applet

- 按下“Build trees”生成 UPGMA。
- 编辑距离矩阵的字段(只有下面的字段可以被编辑，当重新生成树时，这个矩阵自动被转换成对称矩阵)。
- 如果想改变距离矩阵的大小，编辑“Sequence count”字段并按下“New input size”。
- 按下“Random data”用随机距离数据来填充矩阵。
- 出于安全原因，当在浏览器中运行 applet 时，“Print tree”按钮不起作用。

层次聚类方法总结

- 无需事先指定簇的数目。
- 层次本质很好的反映了人类对某些领域认识的直觉。
- 可伸缩性不好：时间复杂性至少为 $O(n^2)$ ，其中 n 是所有对象的数量。
- 和任何启发式搜索算法一样，局部最优是一个问题。
- 对结果的解释具有主观性。

11.2 划分聚类

k-均值聚类

k-均值 (MacQueen, 1967) 是用来解决著名的聚类问题的最简单的非监督学习算法之一。该过程遵循一个简易的方式，将一组数据划分为预先设定好的 k 个簇。其主要思想是为每个簇定义一个质心。设置这些质心需要一些技巧，因为不同的位置会产生不同的聚类结果。因此，较好的选择是使它们相互之间尽可能远。接下来将数据中的每个点与距它最近的质心联系起来。如果再无数据点未与相关质心相联，那么第一步就结束了，早期聚合过程也相应完成。此时，我们根据上一步所产生的结果重新计算 k 个质心作为各个簇的质心。一旦获得 k

个新的质心, 我们需要重新将数据集中的点与距它最近的新质心进行绑定。一个循环就此产生。作为循环的结果, 我们发现 k 个质心逐步改变它们的位置, 直至位置不再发生变化, 即质心不再移动为止。

k -均值非常适用于产生球状簇。 k -均值方法是数值的、非监督的、非确定的、迭代的。

最后, 该算法旨在最小化一个目标函数——在此处, 是误差平方函数。目标函数为:

$$J = \sum_{j=1}^k \sum_{i=1}^n \|x_i^{(j)} - C_j\|^2$$

式中, $\|x_i^{(j)} - C_j\|^2$ 为数据点 $x_i^{(j)}$ 到簇中心 C_j 的距离度量, 也指示 n 个数据点与其各自簇中心的距离。

算法包括以下几个步骤:

- 1) 决定 k 的取值。
- 2) 初始化 k 个簇中心(如果需要可以任意设置)。
- 3) 通过把对象分配给最近的簇中心来确定 N 个对象的簇隶属关系。
- 4) 假设上面所得的隶属关系是正确的, 重新估计 k 个簇中心。
- 5) 如果在最后一次迭代中 N 个对象无一再改变隶属关系, 则退出; 否则, 转到第 3 步。

尽管可以证明这个过程总是收敛的, 但是 k -均值算法并不能保证找到对应于最小化全局目标函数的最优解。算法对于随机选取的初始簇中心非常敏感, 可以通过多次执行该算法来减少初始中心敏感的影响。

例子 假设我们现在有 n 个同一类型的样本特征向量 x_1, x_2, \dots, x_n , 并且我们知道它们属于 k 个紧凑的簇($k < n$)。令 m_i 表示属于簇 i 的所有向量的均值。如果簇与簇是良好分离的, 则可以用最小距离分类器来分开它们。也就是说, 如果 $\|x - m_i\|$ 是 k 个距离中最小的, 则我们可以判定 x 属于簇 i 。这表明可以用如下步骤来求 k 个均值:

```

对均值  $m_1, m_2, \dots, m_k$  做初始猜测
Repeat
    用估算出的均值对样本进行分类
    For  $i$  from 1 to  $k$ 
        用簇  $i$  中的所有样本的均值取代  $m_i$ 
    End for
Until 再没有均值发生改变
    
```

如图 11-24 ~ 图 11-28 所示, 给出了一个例子说明均值 m_1 、 m_2 和 m_3 如何向三个簇的中心移动。

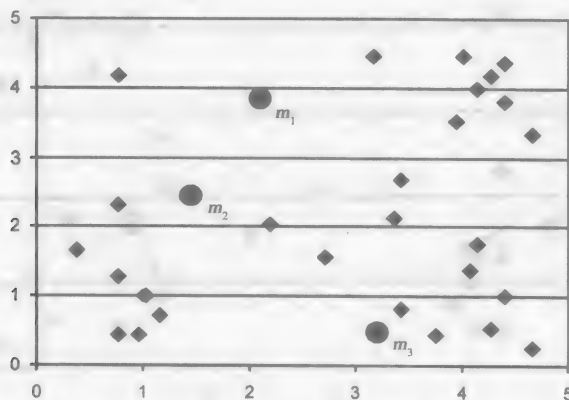


图 11-24 k -均值聚类: 第 1 步

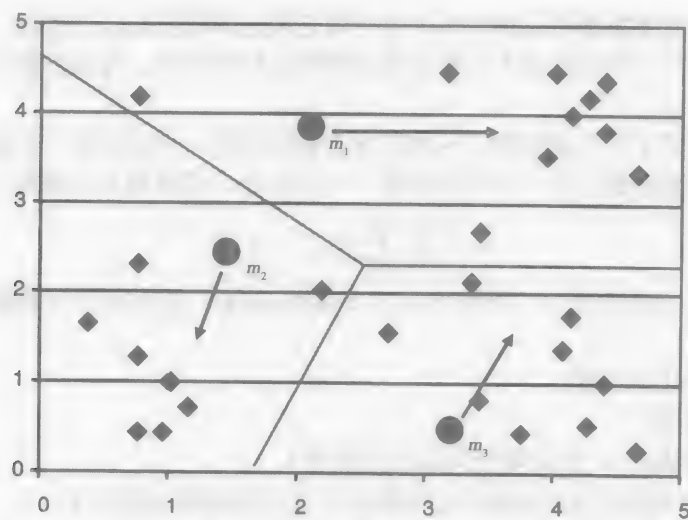


图 11-25 k-均值聚类: 第 2 步

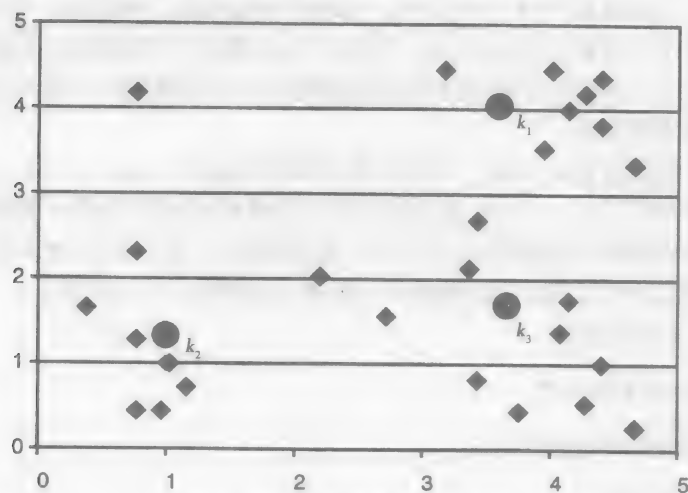


图 11-26 k-均值聚类: 第 3 步

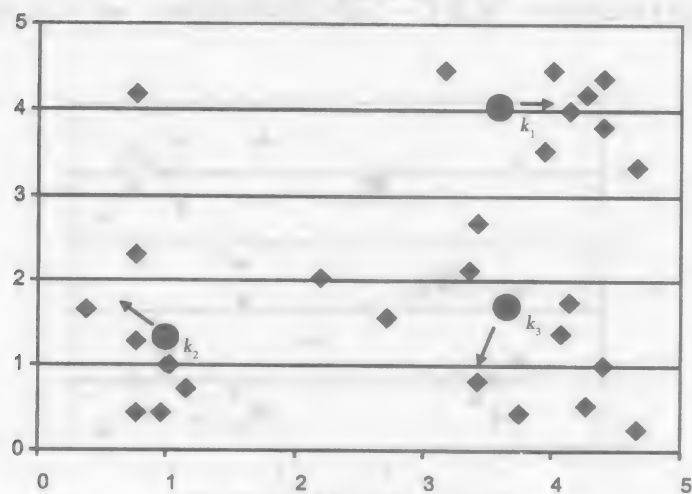


图 11-27 k-均值聚类: 第 4 步

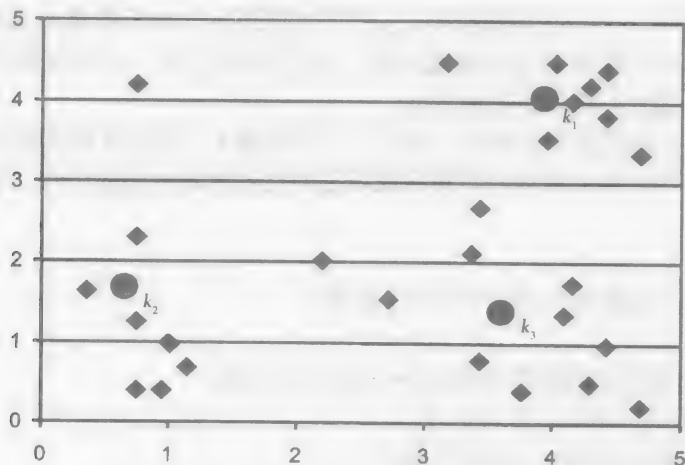


图 11-28 k-均值聚类: 第5步

算法: k-均值, 距离度量: 欧氏距离

评论: 这是一个简化的 k-均值过程。它可以被看作是一个贪心算法, 用于将 n 个样本划分为 k 个簇, 使得所有样本与其簇中心的距离平方和最小。它具有如下缺点:

- 没有指明初始化均值的方法。常用方法是随机选取 k 个样本作为均值。
- 产生的结果依赖于均值的初始值, 经常发生得到次优划分的情况。解决方法是多次尝试不同的初始值。
- 可能发生距离簇中心 m_i 最近的样本集为空的情况, 因此 m_i 将得不到更新。这是一个必须处理的棘手问题, 但是我们忽略该问题。
- 结果依赖于 $\|x - m_i\|$ 的度量单位。一个常用解决方法是用标准差规范化各个变量, 虽然这并非总是可取的。
- 结果依赖于 k 值。

鉴于我们通常无法知道存在几个簇, 最后一个问题是非常棘手的。遗憾的是, 对给定数据集, 也不存在通用理论解决方案来找到它的簇数。一个简单方法是, 采取不同的 k 值, 得到多个运行结果, 比较这些结果, 选择满足给定准则的最好结果。但是需要注意, 因为根据定义, 增加 k 会使误差函数值更小, 但也会增加过分拟合的风险。

该方法的优点:

- 如果变量很大, k-均值比层次聚类的计算速度更快(如果 k 很小)。
- 与层次聚类相比, k-均值可以得到更紧密的簇, 尤其是对于球状簇。

该方法的缺点:

- 难以比较聚类结果的优劣(例如, 不同的初始划分或者 k 值会影响输出结果)。
- 固定的簇数使得很难事先对 k 值做出预测。
- 对于非球状簇效果不好。

不同的初始划分得到不同的最终聚类结果, 使用相同的和不同的 k 值运行程序, 比较得到的结果是有帮助的。

11.3 k-中心点

k-均值和 k-中心点拥有相同的过程。在 k-中心点算法中, 只有样本空间中的数据点可

以作为中心点；然而，在 k -均值算法中，空间中的任何点（接近数据点或数据点本身）都可以成为均值点。通过计算数据点和假设的中心点之间的代价，中心点或者被保留或者被交换，直到那些假设的中心点不再更改为止。

k -中心点是一个典型的划分算法。对于一个给定的 k ，采用该算法的目标是在数据集中寻找 k 个代表，使得把每个对象划归到它最邻近的代表所表示的簇中时，对象和代表的距离和最小。

算法

1) 任意选取 k 个对象作为初始中心点(代表)

2) Repeat

把剩余对象分配到距它最近的中心点所在的簇；

随机选择一个非中心点对象 O ；

计算随机用 O 交换 O_j 的总代价 S ；

如果 $S < 0$ ，则用 O 交换 O_j ，形成新的 k 个中心点的集合；

Until 无变化发生

3) 结束

k -中心点算法工作方式

k -中心点聚类算法的基本策略是，首先在数据中为每个簇随意找一个代表对象，从而在 n 个对象中发现 k 个簇。这些作为代表的对象称为中心点(medoid)，其余对象为非中心点(non-medoid)。计算所有非中心点到每个中心点的距离，并将所有非中心点划分到距它最近的簇(即距离最小的中心点)。只要聚类结果可以被改善，便不断地用非中心点替代中心点。聚类质量通过代价函数来评价，该代价函数反映了对象和它所属簇的代表之间的平均相异性。

1) 任意选择 k 个代表对象。

2) 计算每对代表对象 O_i 和非代表间 O_h 的 TC_{ih} 。

3) 选择满足 $\min(O_i, O_h, TC_{ih})$ 的 O_i, O_h 对，如果最小的 TC_{ih} 为负，用 O_h 代替 O_i ，返回到第 2 步。

4) 否则，为每个非代表对象寻找最相似的代表对象。

11.4 现代聚类方法

如前节所述，传统的聚类方法可以分为两类。现代聚类方法可以分成五类，其中也包括传统的层次方法和划分方法。

1) 层次方法

2) 划分方法

3) 基于密度的方法

4) 基于网格的方法

5) 基于模型的方法

这里，我们将简要介绍每一类的主要特征，及其近期文献中出现的每类的主要算法。

1. 层次方法

层次方法相继地将相对较小的簇合并为较大的簇，或者分裂较大的簇。算法的结果为一棵聚类树，称为树状图(建立给定数据对象的层次分解)。根据层次分解的形成方式，该方法可以是凝聚的(自底向上，由单个对象形成单独的分组开始)或分裂的(自顶向下，由包含所有对象的簇开始)。层次聚类是不可逆的，一旦合并或者分裂完成，便不能恢复到之前的

状态。为了弥补合并或分裂的僵硬性,可以通过在每次层次划分时分析对象间的连接关系,整合其他聚类技术(如迭代重定位)来改善层次凝聚的质量。

例子:

- BIRCH(Balanced Iterative Reducing And Clustering Using Hierarchies, 利用层次方法的平衡迭代归约和聚类): 它使用一种称为 CF-树的层次数据结构,以增量和动态方式来划分到来的数据点。CF-树是一种用来存储聚类特征的高度平衡树。它基于两个参数:分支因子 B 和阈值 T , 每个簇的直径必须小于 T 。
- CURE(Clustering Using REpresentatives, 使用代表对象聚类): 它用一定数量的点来表示每个簇, 这些点是通过选择分散良好的点而产生的。然后对每个簇按照指定量向簇质心进行收缩。它采用随机抽样和划分聚类来处理大规模数据库。
- ROCK 方法: 用来对布尔数据和分类数据聚类的鲁棒的聚类算法。它产生点的近邻和连接概念, 并基于它们来度量数据点间的相似性和接近性。

2. 划分方法

该方法试图直接将数据分解成分离的簇。首先创建一个初始 k 划分, 其中参数 k 表示要构造的划分数; 然后利用迭代重定位技术, 即通过移动不同划分中的对象来改善划分质量。该方法适用于在小规模或中等规模的数据集中寻找球形簇。

例子:

- k -均值: 每个簇用簇内对象的均值来表示。将所有点分配给与其距离最近的簇; 然后重新计算每个簇的质心。不断重复这一过程, 直到簇质心不再改变为止。
- PAM(Partitioning Around Medoids, 围绕中心点划分): 每个簇由距它中心最近的对象所代表。开始, 算法为所有 k 个簇分别找到一个对象作为其中心点, 随后将剩余所有对象分配到和它最相似的中心点所代表的簇中。将中心点与未被选作中心点的对象进行交换, 直到没有未被选到的对象能够担任中心点为止。
- CLARA(Clustering LARge Application, 大型应用聚类): 它是在数据集的子集上实施 PAM 算法。它从数据集中抽取多个样本子集, 利用 PAM 算法对每个样本子集进行处理, 然后选择这些样本子集的最佳聚类结果。
- CLARANS(Clustering Large Application based on RANdomised Search, 基于随机搜索的大型应用聚类): 寻找一个图, 图中的每个节点是一个潜在的解, 即 k 个中心点的集合。它选择节点, 并与用户事先定义好数目的近邻进行比较, 寻找局部最小值, 然后移动它到近邻节点。如果找到局部最优解, 它开始重新随机选择节点并寻找新的局部最优解。
- k -众数(k -mode): 基于 k -均值算法, 但是主要针对分类数据聚类。

3. 基于密度的方法

该方法根据密度条件对邻近对象分组形成簇。簇的增长或者根据邻域密度, 或者根据特定的密度函数。

例子:

- DENCLUE(DENSity-based CLUstEring, 基于密度的聚类): 将整个空间的密度定义为所有数据点影响函数之和, 通过确定密度吸引子(attractor)来识别簇。
- DBSCAN(Density-Based Spatial Clustering of Application with Noise, 具有噪声的基于密度的空间聚类): 对于簇中的每个点, 其给定半径内的近邻数至少超过某个给定值。该算法可以处理噪声数据, 并能发现任意形状的簇。
- OPTICS(Ordering Points To Identify the Clustering Structure, 点排序识别聚类结构): 该

方法计算簇增长顺序,以便进行自动或交互式聚类分析。

4. 基于网格的方法

该方法将空间量化为数量有限的单元以形成网格结构,然后在网格结构上进行聚类。

例子:

- STING (STatistical INformation Grid-based method, 统计信息基于网格的方法): 用层次结构将空间区域分割成一些矩形单元。遍历整个数据集, 计算各个单元内对象的数值特征的统计参数, 然后创建一个网格单元的层次结构来表示不同层次上的聚类信息。
- 小波聚类 (Wave cluster): 基于信号处理技术将空间数据转换到频率域。首先通过对数据空间施加多维网格结构来概括数据, 每个网格单元概括了落入该单元的数据点的信息。之后采用小波变换来转换原始特征空间。它通过在变换域内寻找密集区域来确定簇。

5. 基于模型的方法

该方法为每个聚类假设一个模型, 寻找数据与模型的最佳拟合。典型的基于模型的方法包括统计方法 (COBWEB、CLASSIT、AutoClass) 和神经网络方法 (竞争学习和自组织映射)。

在接下来的几节中, 将讨论一些流行的聚类算法。考虑到处理器所面临的困难或者不同类型数据, 产生了为数众多的聚类算法。

11.5 BIRCH

Zhang 等人 [13] 提出了 Birch (Balanced Iterative Reducing and Clustering) 算法来对大规模数据集进行聚类。它是一种增量算法, 可以根据可用内存大小调整内存需求。作者使用了称为聚类特征和 CF 树的概念。

聚类特征 (Clustering Feature, CF) 是一个三元组, 概括了子簇信息。假定子簇中包含 N 个 d 维点: $\{\vec{X}_i, i = 1, 2, \dots, N\}$, CF 定义为:

$$CF = (N, \vec{LS}, \vec{SS})$$

其中 N 表示子集内点的数目; \vec{LS} 表示 N 个点的线性和 $\sum_{i=1}^N \vec{X}_i$, \vec{SS} 为数据点的平方和 $\sum_{i=1}^N X_i^2$ 。

例如, 考虑一个由二维点组成的集合 $\{(3, 4), (2, 6), (4, 5), (4, 7), (3, 8)\}$, 其形成的子簇如图 11-29 所示。

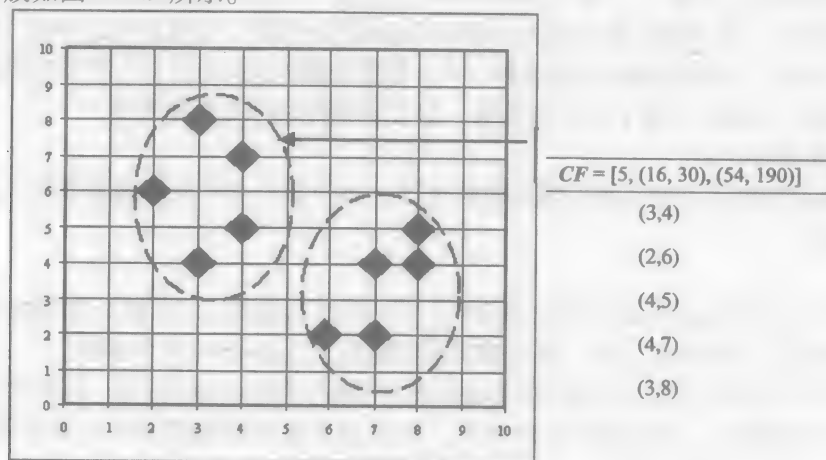


图 11-29 5 个数据点的聚类特征 (CF)

这里

$$N=5$$

$$\vec{LS} = (3, 4) + (2, 6) + (4, 5) + (4, 7) + (3, 8) = (16, 30)$$

$$\vec{SS} = (3^2, 4^2) + (2^2, 6^2) + (4^2, 5^2) + (4^2, 7^2) + (3^2, 8^2) = (54, 190)$$

$$CF = [5, (16, 30), (54, 190)]$$

聚类特征足以用来计算簇距离, 并且它们创建了有效的信息存储方法, 因为它们概括了子簇的信息而不是存储所有的数据点。

CF 树是一棵具有两个参数的平衡树: 分支因子 B 和阈值 T 。分支因子指定了子节点的最大数目。阈值参数指定了存储在叶节点的子簇的最大直径。改变该阈值可以改变树的大小。非叶节点存储的是它的子女节点的 CF 之和, 因此, 它们概括了它们的子女的信息。CF 树是随着数据点的插入而动态创建的, 因此该方法是增量的。点被插入到最接近的叶节点(子簇)中。如果插入后存放在叶节点子簇的直径超过阈值, 那么叶节点和可能的其他节点被分裂。插入新点后, 该点的信息向树根的方向传递。可以通过改变阈值来改变 CF 树的大小。如果存储 CF 树所需的内存大小超过主存储器容量, 那么可以指定一个更大的阈值并重建 CF 树。我们通过一个例子来说明这个过程。当添加一个新的点时, 所得到的树如图 11-30 所示。我们假设存储器不足以存储整棵树。如果是这样的话, 算法合并最接近的树并得到如图 11-31 所示的树。

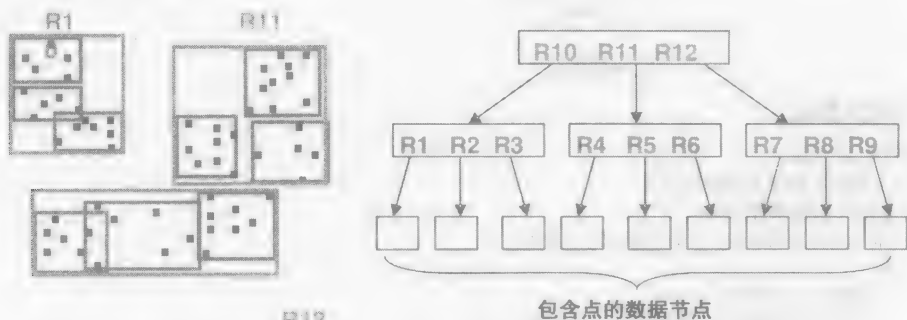


图 11-30 合并前的树

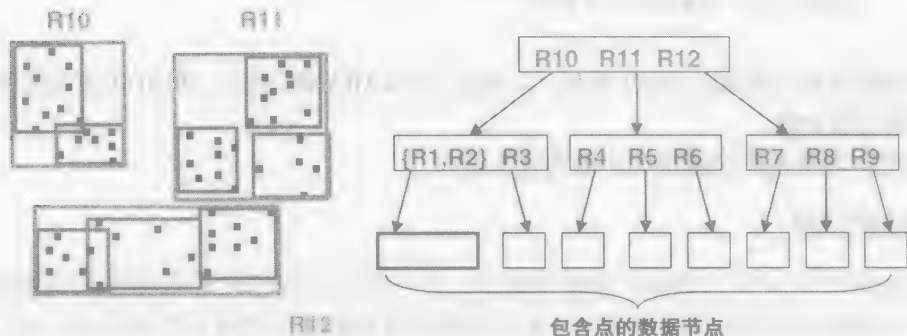


图 11-31 合并后的树

在重建过程中, 通过利用老树的叶节点来重新构建一棵新树, 因而树的重建过程不需要访问所有点。所以, 构建 CF 树只需访问数据一次即可。

最后, 如果我们在算法中指定所要生成的簇的数目, 它合并最近的子树并构造所需数目的簇。在图示的例子中, 如果我们需要两个簇, 则我们会得到如下的两个簇(如图 11-32 所示)。

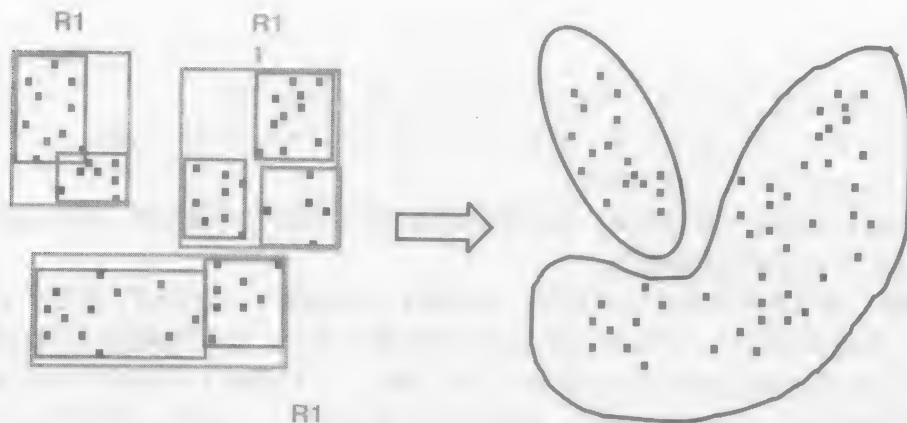


图 11-32 子簇合并

Birch 算法

输入:

$D = \{\vec{x}_1, \vec{x}_2, \dots, \vec{x}_N\}$ // 元素集
 T // CF 树构建阈值
 B // 分支因子

输出:

K // 簇的集合

Birch 聚类算法

```

For 每个  $X_i \in D$  do
  为  $X_i$  的插入确定正确的叶节点
  If 不违反阈值条件 then
    将  $X_i$  添加到簇中, 并且更新 CF 三元组;
  Else
    If 有插入  $X_i$  的空间 then
      作为单个簇插入, 并更新 CF 三元组;
    Else
      分裂树叶节点, 并重新分布 CF 特征;
  
```

总结

- 与前面的基于距离的方法(例如, k -均值和 CLARANS)相比, BIRCH 更适合处理超大规模的数据集。
- BIRCH 可以在任何给定大小的内存下运行。

11.6 DBSCAN

聚类算法 DBSCAN[3] 依赖于簇密度概念, 并且用来发现任意形状的簇和识别噪声。根据对象的空间和非空间的属性, DBSCAN 可以聚类点对象和在空间上扩展对象。基于密度的聚类基于簇的密度高于周边环境这一事实。换言之, 簇是由对象密度低的区域所分离的对象密集区域。图 11-33 显示了一个典型的具有变化的密度和形状的二维簇。

传统的聚类算法, 如 k -均值和 k -中心点, 假设簇本质上是球形的, 并且需要给定簇的数目作为输入。因此, 在很多情形下, 它不能发现数据中的自然分组。图 11-34 显示了 k -中心点对人为生成数据聚类的结果。

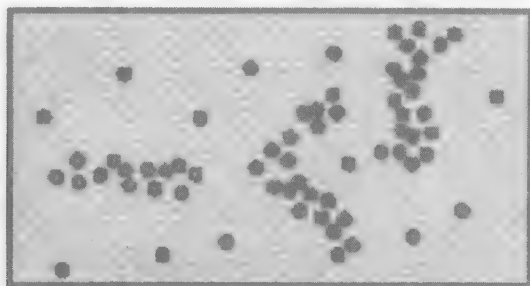


图 11-33 不同密度的聚类

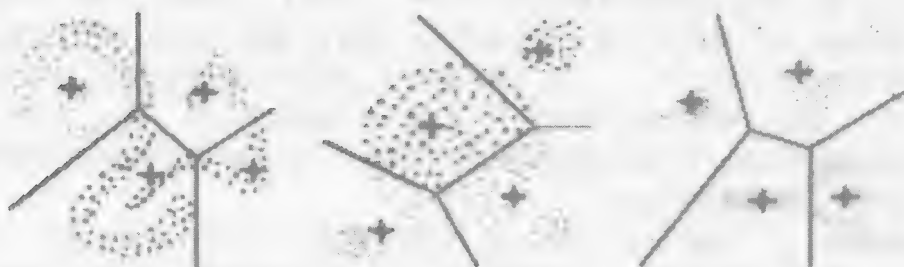


图 11-34 k-中心点对不同数据集的簇结果

对 DBSCAN 的基本概念的初步认识:

- 1) 对于簇中的任意一个点, 它周围局部点密度必须超过某阈值。
- 2) 簇中的点在空间上是相互关联的。

任意点 p 的局部点密度由两个参数定义。它们是用用户定义的参数。参数在聚类时作为输入和数据一起提供。这些参数是:

- 1) 点 p 的邻域的 ε -半径: 给定 ε , 我们可以找到落在点 p 半径 ε 内的近邻的数量。该数量依赖于半径 ε 。我们用 $N_\varepsilon(p)$ 表示落在点 p 的 ε -半径内的点的集合。该集合可以表示为:

$$N_\varepsilon(p) = \{q | q \text{ 在数据集 } D \text{ 中, 使得 } distance(p, q) \leq \varepsilon\}$$

- 2) $MinPts$: 给定邻域 $N_\varepsilon(p)$ 包含的点的最小数目。(这个数以一种方式在算法中使用, 以决定点 p 是在簇的核心部分还是边界点或噪声。)

11.6.1 DBSCAN 算法的概念

核心对象: 在其 ε 邻域内具有至少 $MinPts$ 个点 (即以给定对象为中心划一个半径为 ε 的圆, 这个圆至少包含 $MinPts$ 个点, 这个给定的对象就是核心对象)。例如, 如果 ε 为 2 个单位并且 $MinPts = 5$, 为了满足核心对象条件, 一个对象应当在它的以 ε 为半径的邻域中至少包含 5 个点 (对象) 作为它的近邻 (包括它自己)。如图 11-35 所示。

边界对象: 在核心对象的 ε 邻域内, 但不满足核心对象条件的对象即为边界对象 (border object)。例如, 如果 ε 为 2 个单位而 $MinPts = 5$, 并且 P 在某核心对象的 ε 邻域内, 则在它的 $\varepsilon = 2$ 的邻域中包含近邻少于 5 个对象时即为边界对象。

直接密度可达的: 点 P 是由点 Q 关于两个参数 (ε , $MinPts$) 直接密度可达的, 如果满足

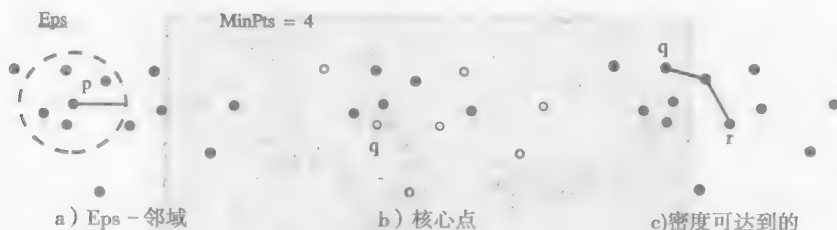


图 11-35 基于密度聚类的概念

1) P 属于点 Q 的 ε 邻域。

2) 在点 Q 的 ε 邻域所包含的点数量大于 $MinPts$, 即 $|N_{\varepsilon}(Q)| \geq MinPts$ (核心对象条件)。

密度可达的: 点 P 是由点 Q 关于两个参数 (ε , $MinPts$) 密度可达的, 当且仅当存在由 Q 开始的一连串的点 $P_1, P_2, P_3, \dots, P_n = P$, 满足 P_{i+1} 是从 P_i 直接可达的。开始点 Q 应该是一个核心点。例如, 有两个远距离的点 P, Q 和一些中间点 $P_1, P_2, P_3, \dots, P_n$, 如果 P 到 P_1 是直接密度可达的, P_1 到 P_2 是直接密度可达的, \dots , 直到 P_n 到 Q 也是直接密度可达的, 那么点 P, Q 是密度可达的。

密度相连的: 点 P 与点 Q 是关于 $\varepsilon, MinPts$ 密度连接的, 当存在点 O , 使 P, Q 都是由点 O 关于 $\varepsilon, MinPts$ 密度可达的。也就是说, P, Q 必须是由任意核心点密度可达的, 而 P, Q 不一定是核心点。如图 11-36 所示。

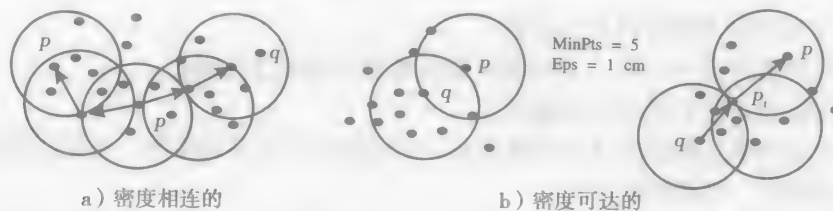


图 11-36 密度相连和密度可达的含义

噪声点: 如果说点 P 既不是核心对象也不是任何其他点密度可达的, 那么它就是一个噪声点。

11.6.2 DBSCAN 的基本概念和算法

1) 密度相连集中的每个对象都是密度可达的。

2) 选择任意一点 P 。

3) 如果点 P 未被分类, 那么检查核心点条件。

4) 如果该点为核心点, 找到所有由 P 关于 $\varepsilon, MinPts$ 密度可达的点。

5) 用这些点形成一个新的簇, 给每个点分配一个簇 ID (同一簇中的所有点拥有相同的簇 ID)。

6) 如果点 P 为边界点 (即没有从 P 密度可达的点), 则继续访问数据中的下一个点。

7) 继续这个过程, 直至处理完所有点为止。

这个过程用那些密度可达的点创建簇, 并隔离远离的噪声点。

11.6.3 算法

```

For 每个  $o \in D$  do
  If  $o$  还未被分类 then
    If  $o$  是核心对象 then
      寻找所有可以由  $o$  密度可达的对象, 并指派为一个新簇。
    Else 将  $o$  划归为噪声对象。

```

例子 考虑 $\varepsilon = 3$ 个单位, $MinPts = 4$ 来理解 DBScan 算法如何工作。数据如图 11-37 所示。

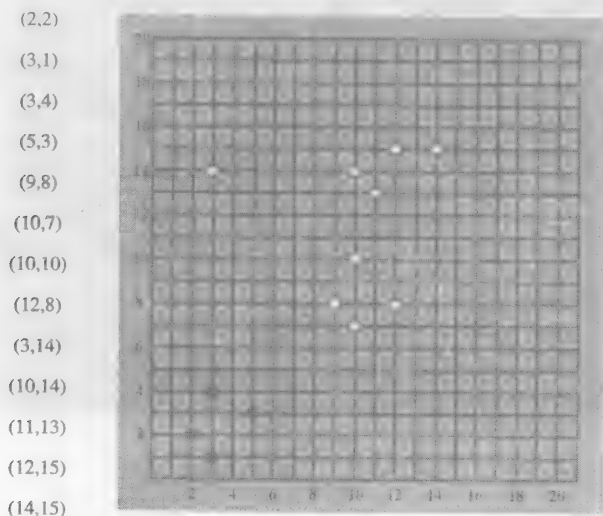


图 11-37 关于数据点的图

描述:

- 1) 随机选择一个点, 比如说 $(3, 1)$ 。设 $NP = \{(3, 1)\}$ 。
- 2) 由那个点检查其近邻点是否来自它的 ε 邻域。
- 3) 让我们尝试最近的点 $(2, 2)$, 这两个点之间的距离是 2 个单位。因此, 它在 ε 邻域之内, 我们把它加入到一个新簇, 并把它加入到 NP 集。现在 $NP = \{(3, 1), (2, 2)\}$ 。
- 4) 现在我们再尝试下一个最近的点 $(3, 4)$, 距离为 3 个单位, 等于 ε 。把它加入到簇中。新的 $NP = \{(3, 1), (2, 2), (3, 4)\}$ 。
- 5) 接下来尝试点 $(5, 3)$, 它们的距离是 4 个单位, 大于 ε 。因此, 它在点 $(3, 1)$ 的 ε 邻域之外, 但是它也可以被加入到簇中, 只要它在该簇的某个点的 ε 邻域内。
- 6) 寻找点 $(3, 1)$ 的 ε 邻域中的其他点。
- 7) 如果找不到新的点, 选择簇中的其他点 (比如点 $(2, 2)$ 或者 $(3, 4)$), 并重复步骤 1~6。
- 8) 聚类落在 NP 中的每个点的 ε 邻域内的所有点。
- 9) 在图 11-37 中, 由点 $(3, 1)$ 开始共有 4 个点形成一个簇, 即 $NP = \{(3, 1), (2, 2), (3, 4), (5, 3)\}$ 。
- 10) 对于簇中的每个点, 如果在其 ε 邻域内找不到新的点, 那么任意选择其他点重复上面过程。

11) 如果所选的点不在任何簇的 ε 邻域内, 则该点为噪声点。

12) 在图 11-37 中, 共有三个簇和一个噪声点。图 11-38 显示了被聚类的点。

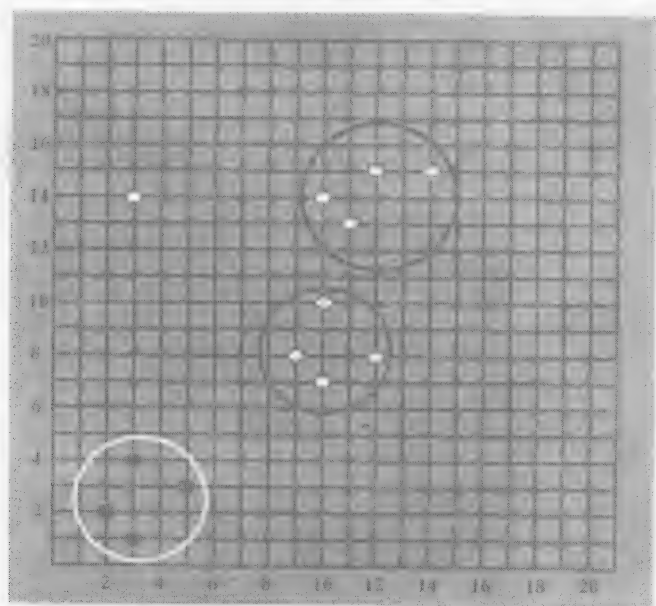


图 11-38 聚类后的图

11.6.4 DBSCAN 算法的优点

- 1) 形成的簇可以具有任意的形状和大小。
- 2) 可以自动确定形成的簇数目。
- 3) 可以分离簇和环境噪声。
- 4) 可以被空间索引结构所支持。
- 5) 效率高, 即使对大数据集也是如此。
- 6) 一次扫描数据即可完成聚类。

11.7 OPTICS

11.7.1 引言

Optics 算法[4]是用于聚类分析的新算法, 它并不显式地产生一个数据集的聚类, 取而代之的是建立一个增广的数据库排序来表示它基于密度的聚类结构。这个簇序(cluster ordering)包含的信息等同于对应一系列参数设置的基于密度的聚类。

Optics 的工作原理与用无限数量的距离参数 ε' 扩展的 DBSCAN 算法类似, 该参数 ε' 小于“生成距离” ε 。唯一不同的是, 我们不指定簇成员关系, 取而代之的是, 我们存储对象被处理的顺序以及可以被扩展的 DBSCAN 算法用来指定簇成员的信息。该信息对每个对象来说包含两个值: 核心距离和可达距离(即 OPTICS 产生一个数据库排序并存储每个对象的核心距离和适当的可达距离)。

11.7.2 OPTICS 算法的动机

1) 聚类算法需要输入参数值。该参数值很难确定, 对于包含高维值的现实世界数据集尤其如此。

2) 算法对于参数值敏感, 对于同一数据集, 即使参数设置稍微不同也可能导致划分结果差异显著。

3) 高维真实数据集通常具有倾斜的分布, 因此不能用一个采用唯一全局参数设置的聚类算法来揭示。

11.7.3 OPTICS 采用的概念

核心距离: 对象 P 的核心距离是 P 与其 ε 邻域内使得点 P 以 ε' 为邻域满足核心对象条件的对象的最小距离 ε' 。(如果 P 不是核心对象, 则 P 的核心距离无定义。——译者注)

$\text{Core-distance}(P) = \text{使得 } P \text{ 是核心对象的最小距离 } \varepsilon' \leq \varepsilon$

可达距离: 对象 P 关于另一个对象 O 的可达距离是 P 和 O 之间距离与 O 的核心距离中的最大者。

$$\text{Reachability-distance}(P, O) = \max\{P \text{ 和 } O \text{ 之间的距离, } O \text{ 的核心距离}\}$$

11.7.4 OPTICS 算法

```
// 按可达距离排序的 ControlList
For 每个 Object  $\in D$  do
// 开始, 所有对象  $O$  的  $O.\text{processed} = \text{false}$ 
If Object. $\text{processed} = \text{false}$  then
    将 (Object, "?") 插入到 Control List;
While Control List 非空 do
    从 ControlList 中选择第一个元素 ( $O, r\_dist$ );
    检索  $N_\varepsilon(O)$  并确定  $C\_dist = \text{core-distance}(O)$ ;
    设置  $O.\text{processed} = \text{true}$ ;
    将 ( $O, r\_dist, c\_dist$ ) 写入文件;
    If 对于某个距离  $\leq \varepsilon, O$  是核心对象 then
        For  $p \in N_\varepsilon(O)$  and  $p.\text{Processed} = \text{false}$  do
            确定  $r\_distp = \text{reachability-distance}(p, O)$ ;
            If ( $p, \_$ )  $\in$  ControlList then
                将 ( $p, r\_distp$ ) 插入 ControlList;
            Else if ( $p, \text{old\_r\_dist}$ )  $\in$  ControlList and  $r\_distp < \text{old\_r\_dist}$  then
                用 ( $p, r\_distp$ ) 更新 ControlList 中的 ( $p, \text{old\_r\_dist}$ )
```

让我们考虑一个例子。如图 11-40 所示, 现有用 $A \sim T$ 标记的 20 个点。假设 $\varepsilon = 44$, $\text{MinPts} = 3$ 。

我们随机选择一个点并插入到控制列表 Control list。处理第一个点: 由点 A 开始, 从数据库 D 中寻找 A 的 ε 邻域点, 结果为 B 和 I 。因此, 连同 A 点有 3 个点在它的 ε 邻域中。

A 是核心点。设 $\langle AI \rangle = 38$, $\langle AB \rangle = 40$ 。 A 的最少点近邻[⊖](第 3 个近邻)是点 B 。因

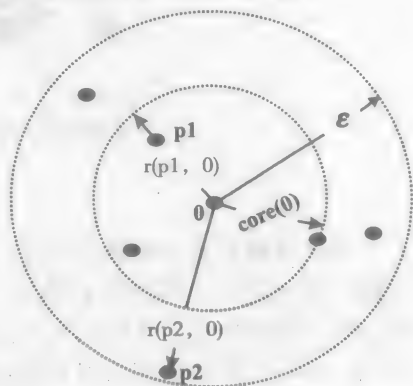


图 11-39 对于 $\text{MinPts} = 4$, 核心距离 $\text{Core-distance}(O)$, 可达距离 $r(p_1, O)$, $r(p_2, O)$

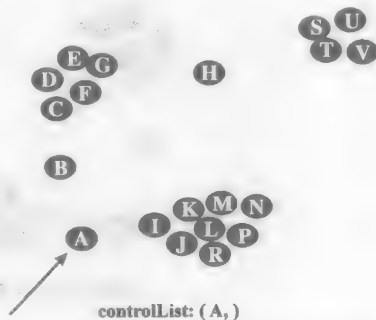


图 11-40 OPTICS 算法所用的数据点

⊖ 点 P 的最少点近邻是使得 P 成为核心对象的最近的近邻。——译者注

此, A 的核心距离是 40。指定点 A 的核心距离属性值为 40, 是否处理过的属性 $processed = true$ 。将 A 写入文件, 将它从控制列表中删除。将 40 指定为点 B 和 I 的可达距离, 并将它们放进控制列表。对于在点 A 的核心距离内的所有点, 它的可达距离就是核心距离本身。因此, I 的可达距离是 40 而不是 38。根据对象的可达距离属性值对控制列表排序。令当前顺序为 $B(40), I(40)$ 。对于第一个点, 可达距离未定义。因此, 在图 11-41 中, 我们用“?”表示点 A 的可达距离未定义。

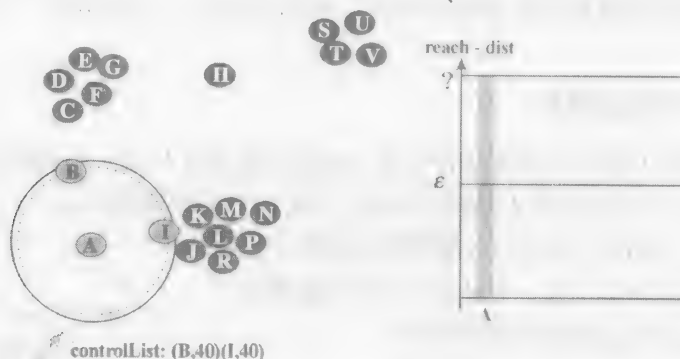


图 11-41 处理第一个点

处理控制列表中的第二个点: 从控制列表中取出第一个点 B , 它就是当前要处理的点。寻找它的 ε 近邻, 得到点 C 和 A 。设 $\langle BC \rangle = 38$, $\langle BA \rangle = 40$ 。 B 是一个核心点 (即以 B 为中心在 ε 邻域内包含 3 个点), 它的核心距离是 40 (因为从 B 到 ε 邻域内的这些点的距离为 0, 38, 40)。把该值作为 B 的核心距离属性, 将它的是否处理过属性设为 $true$, 并添加到文件中。从控制列表中将其删除。在 B 的两个近邻中, 点 A 已经被处理过, 由点 B 到 C 的新的可达距离是 40。由于没有可达距离被赋予点 C , 值 40 被分配给点 C , 并加入到控制列表。对控制列表排序, 并得到当前顺序 $(I, 40), (C, 40)$ 。如图 11-42 所示。

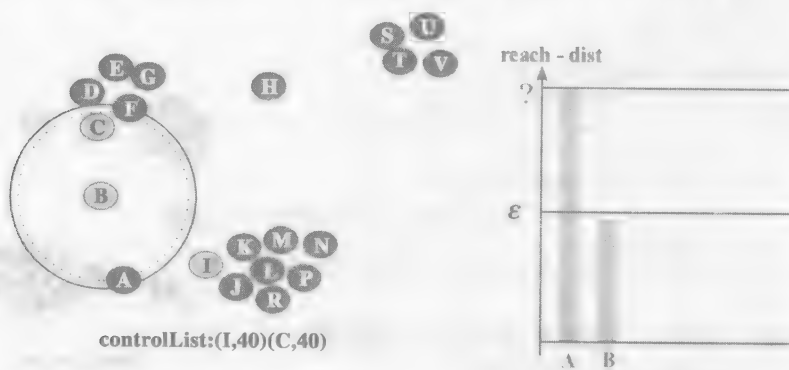


图 11-42 处理第 2 个点 B

处理控制列表中的第三个点: 从控制列表中取出点 I 。寻找它的 ε 近邻。设 A, K, J, L, M 和 R 为它的 ε 近邻。其中 A 已经处理过。设距点 I 的真实距离分别为: $\langle IJ \rangle = 19$, $\langle IK \rangle = 20$, $\langle IL \rangle = 3$, $\langle IM \rangle = 40$, $\langle IR \rangle = 43$, $\langle IA \rangle = 38$ 。最少点近邻是 K 。因此, I 的核心距离为 20。把 I 输出到文件。由于点 J, K, L, M 和 R 并未被处理, 也未出现在控

制列表中, 将 $(J, 20)$, $(K, 20)$, $(L, 31)$, $(M, 40)$ 和 $(R, 43)$ 加入到控制列表。对列表进行排序, 得到 $(J, 20)$, $(K, 20)$, $(L, 31)$, $(M, 40)$, $(R, 43)$, 如图 11-43 所示。

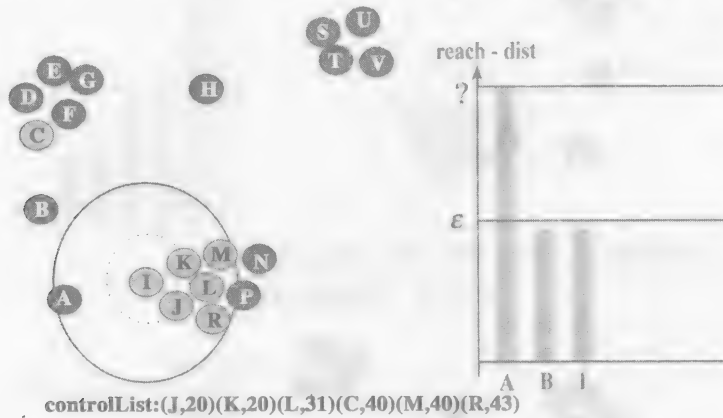


图 11-43 处理第 3 个点 I

处理第四个点: 从控制列表中取出 $(J, 20)$ 。找到它的 ε 近邻点 K, L, R, M, P 和 I 。设点 J 到这些点的实际距离分别为: $JL = 18$, $JI = 19$, $JK = 25$, $JR = 21$, $JM = 30$, $JP = 31$ 。这里第三个最近邻是 I 。由于 $JI = 19$, J 的核心距离为 19, J 是核心点。将 19 作为 J 的核心距离, 并写入文件。现在更新 ε 近邻的可达距离属性。点 K, L, M 和 R 已经在控制列表中。更新它们的可达距离为 $(K, 20)$, $(L, 18)$, $(M, 30)$, $(R, 21)$ 。如图 11-44 所示。

需要注意的是, 由于距 J 的新的可达距离小于之前的可达距离, 因此 L 和 M 的可达距离发生了变化。点 I 已经被处理过, 所以被丢弃。点 P 没有出现在控制列表中, 把 $(P, 31)$ 加入到控制列表。对控制列表排序。排序后的控制列表为:

$(L, 19)$, $(K, 20)$, $(R, 21)$, $(M, 30)$, $(P, 31)$, $(C, 40)$

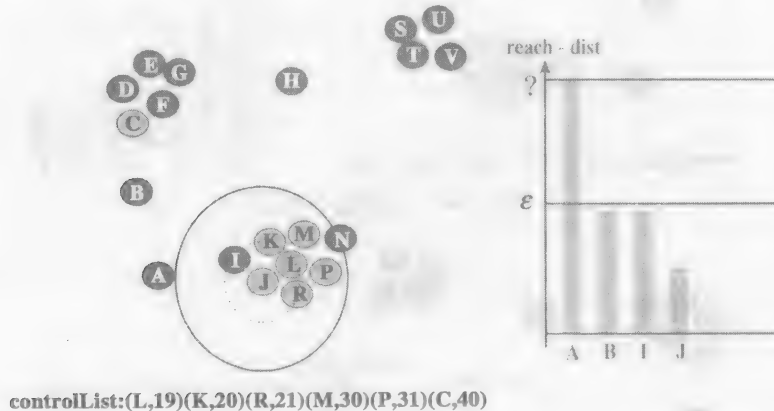


图 11-44 处理第 4 个点 J

继续处理过程: 图 11-45 ~ 图 11-61 表示了算法如何继续为每个点计算可达距离。图中的每个低谷对应于一个簇。

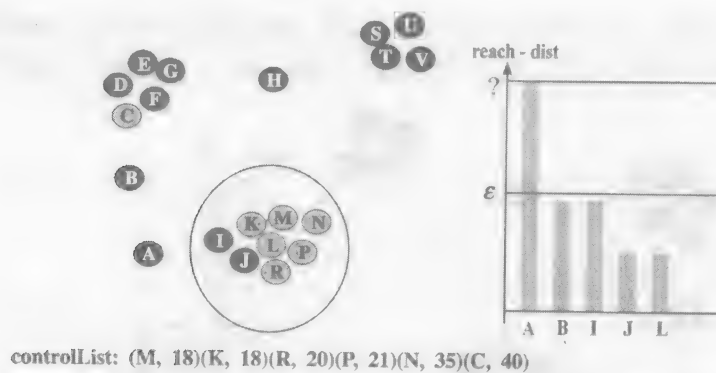


图 11-45 处理第 5 个点 L

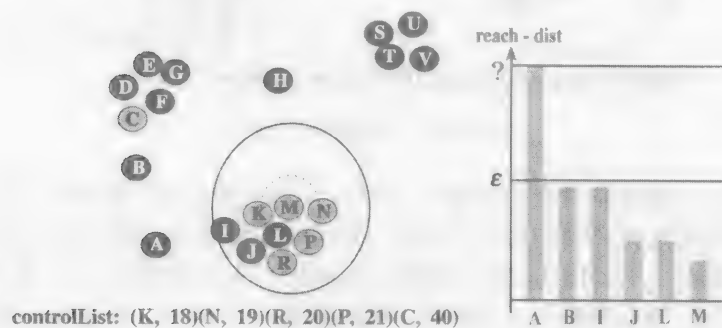


图 11-46 处理第 6 个点 M

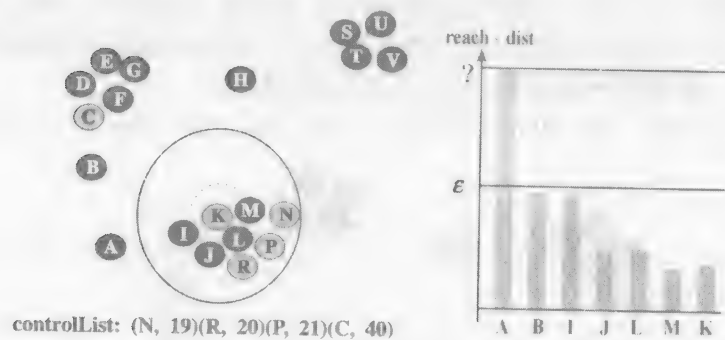


图 11-47 处理第 7 个点 K

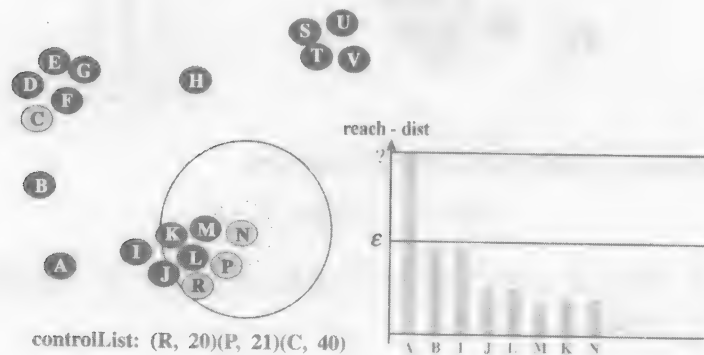


图 11-48 处理第 8 个点 N

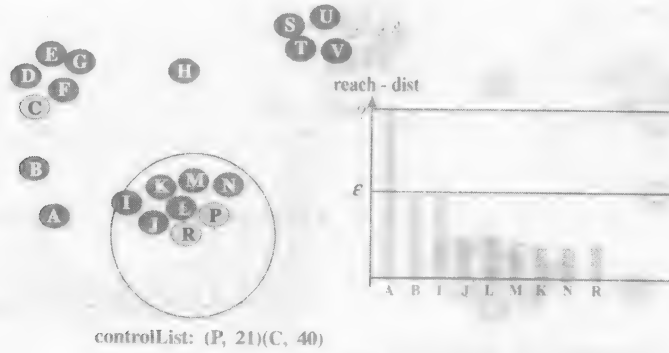


图 11-49 处理第 9 个点 R

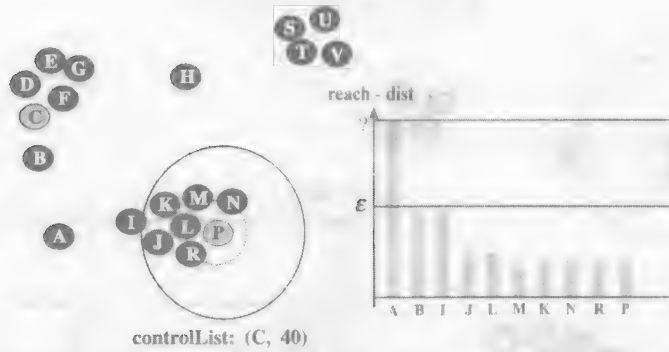


图 11-50 处理第 10 个点 P

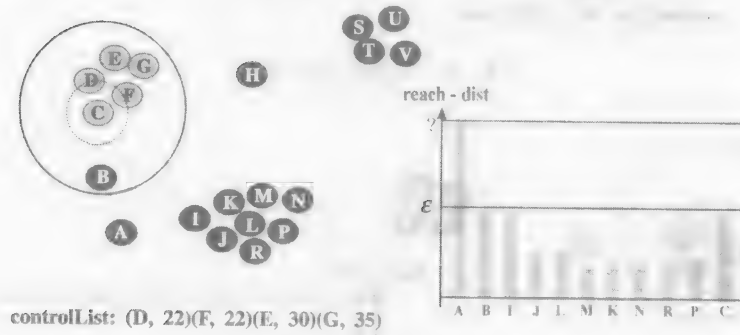


图 11-51 处理第 11 个点 C

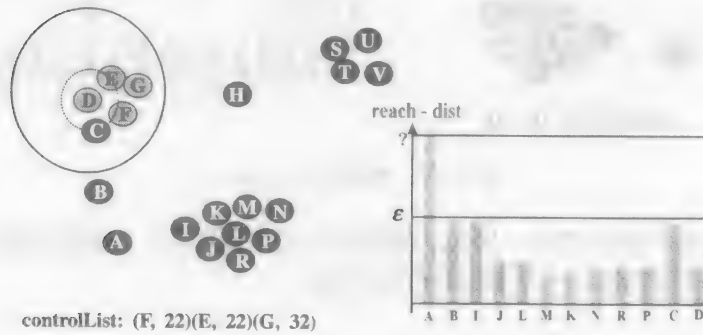


图 11-52 处理第 12 个点 D

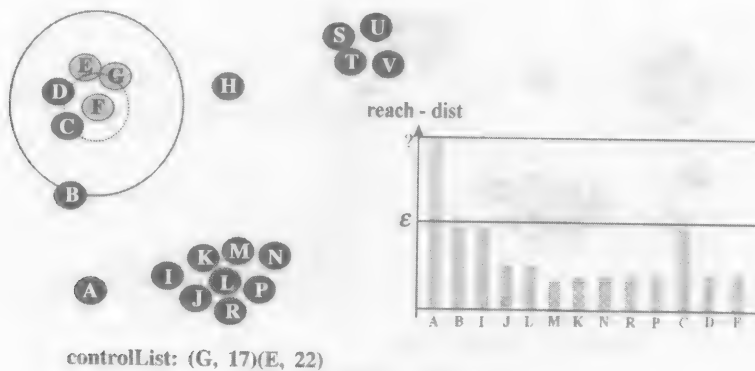


图 11-53 处理第 13 个点 F

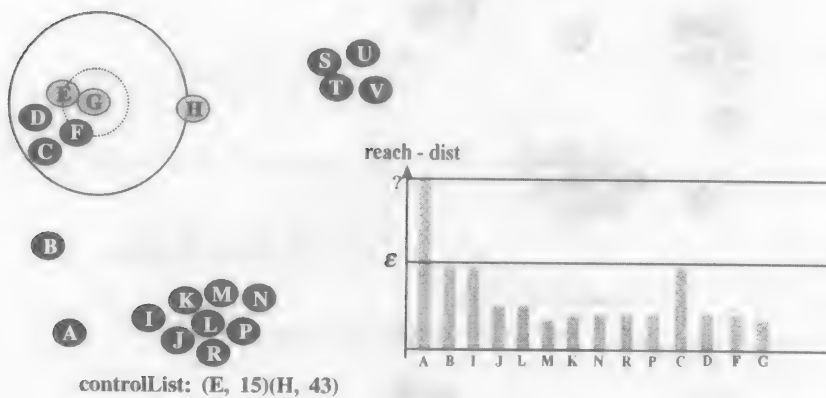


图 11-54 处理第 14 个点 G

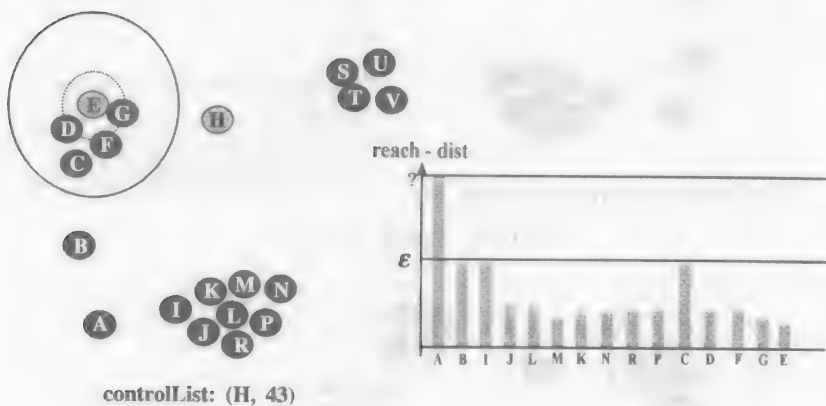


图 11-55 处理第 15 个点 E

在这一点，控制列表变为空。因此，算法进入主循环取点，并检验这些点的已被处理的属性。这一过程持续到找到一个未被处理的点为止。

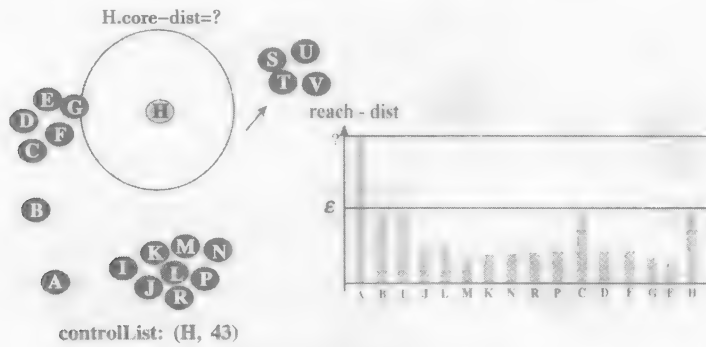


图 11-56 处理第 16 个点 H

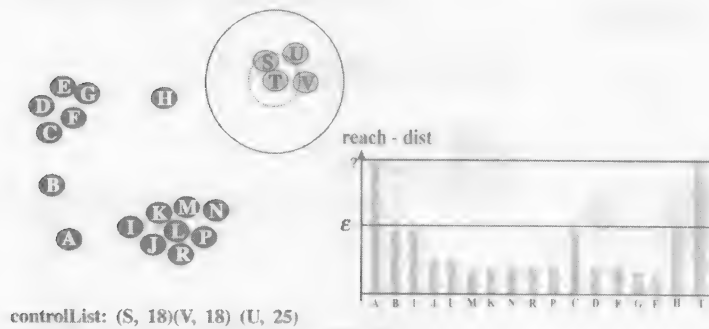


图 11-57 处理第 17 个点 T

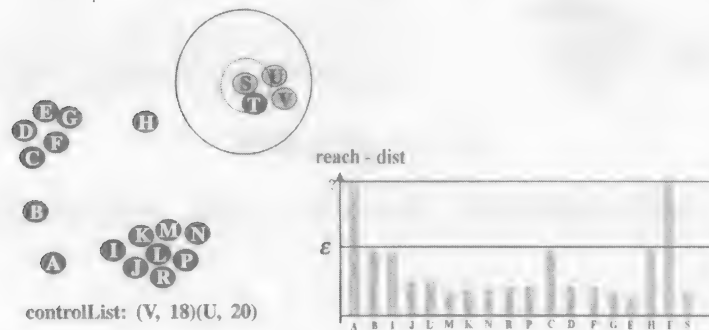


图 11-58 处理第 18 个点 S

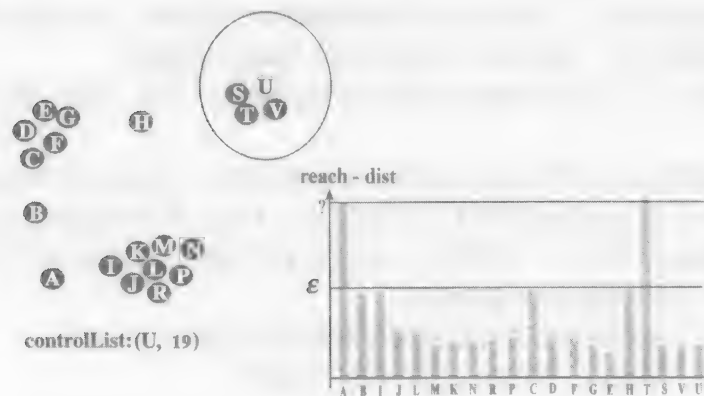


图 11-59 处理第 19 个点 V

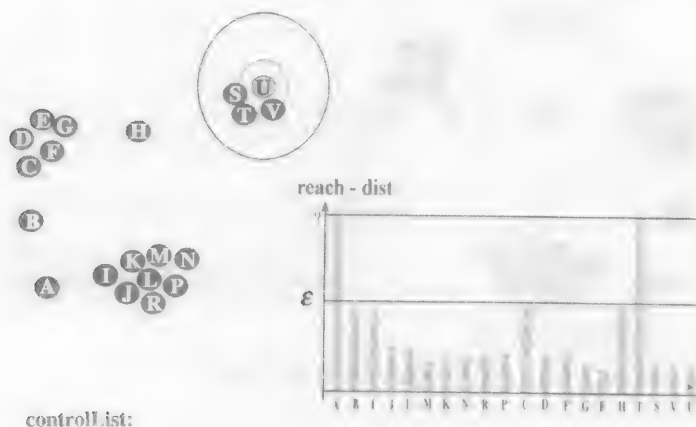
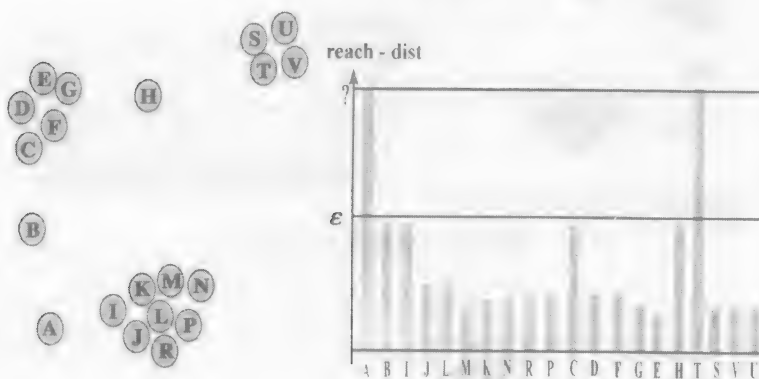
图 11-60 处理第 20 个点 U 

图 11-61 最终的数据簇

最终数据簇：在最后的聚类数据中，算法得到三个聚类。在可达距离图中存在三个谷，因此就有三个类。

11.7.5 可达图

数据集的簇序可以用图的形式来表示和帮理解。大体上，如果簇序中的每个对象 O 的可达距离 r 都绘制在图中，则我们可以看到数据集的聚类结构。可达图是用来清晰理解数据结构的非常直观的方法。可达图对于算法的输入参数相当敏感。

图 11-62 描绘了一个非常简单的二维数据集的可达图。注意，簇序的显示是独立于数据集的。

ϵ 的最优值是满足以下条件的最小的值：使得数据库关于参数 ϵ 和 $MinPts$ 的基于密度的聚类，将仅由一个几乎包含数据库所有点的簇组成。于是，所有聚类级的信息将包含在可达图中。在该最优值的周围存在一系列的值，这些值对可达图外观的改变并不明显。对于不同的 $MinPts$ 值，可达图的总体形状基本相似。

图 11-63 显示了数据集具有变化的簇密度和形状的数据集的可达图。

图 11-64 显示了参数 (ϵ , $MinPts$) 设置对簇序的影响。

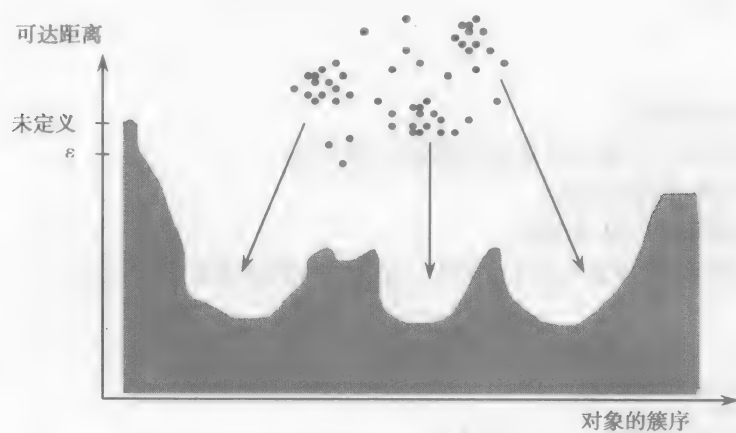


图 11-62 可达图



图 11-63 具有不同尺寸、密度和形状的层次簇的数据集的可达图

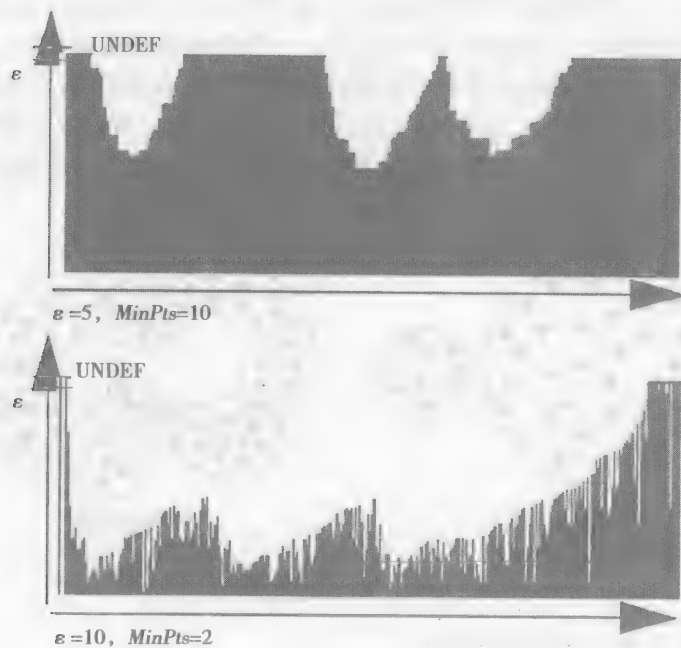


图 11-64 参数设置对簇序的影响

11.7.6 优点

- 1) 无须事先知道簇数。
- 2) 不需要标准的方法或非常鲁棒的参数(OPTICS)。
- 3) 计算簇的完整的层次结构。
- 4) 方法包含很好的结果可视化。
- 5) 随后可以得到“平坦的”划分(例如, 通过截断树状图或可达图)。

11.7.7 缺点

可能不具有良好的可伸缩性。

- 标准方法的运行时间: $O(n^2 \log n^2)$ 。
- OPTICS 的运行时间: 无索引支持情况下 $O(n^2)$ 。

11.8 基于图划分的聚类

11.8.1 加权图划分

聚类可以被看作图划分问题。被聚类的对象可以被看作是图的顶点集 V 。两个对象 x_a 和 x_b (或顶点 V_a 和 V_b) 通过一条权重系数 $W(x_a, x_b)$ 或 $W_{a,b}$ 为正的无向边连接起来。(x_a, x_b) 代表一条从 x_a 到 x_b 的边, 而所有的边形成边集合 E 。边集的基数 $|E|$ 等于所有具有非零相似性的样本对的数目。如果去掉一组边, 图 $G = (V, E)$ 将被划分成 k 对互不相连的子图 $G_n = (V_n, E_n)$, 那么这组边称为边分离器(edge separator)。我们的目的是寻找其边权重之和最小的分离器。然而, 为了得到最小切割, 每个簇中的对象数必须保持大致相等。在这种保持平衡的特殊情况下, 该问题是 NP-困难问题, 称为图划分问题(graph partitioning problem)。

均衡簇是令人满意的, 这是因为每个簇代表一个同等重要的数据部分。然而, 一些自然簇可能在大小上并不相等。通过采用更多的聚类我们可以解决多模类(例如 XOR-问题)并且可以在随后的阶段进行簇合并。这就是 Chameleon 聚类算法[5]所采取的策略。

大多数已有的算法在一些静态案例中效果很好, 但是当数据包含不同形状、密度和大小的簇时, 这些方法便无能为力了。图 11-65 给出了两个这样的簇。

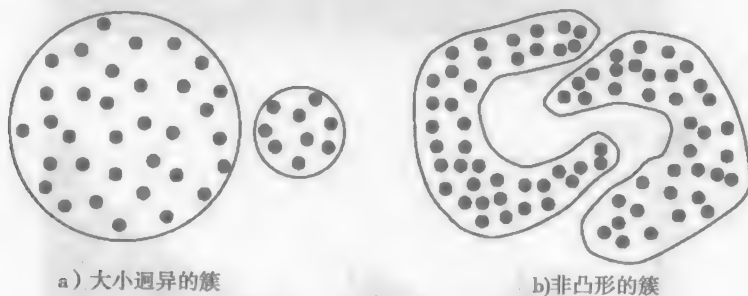


图 11-65 簇的类型

传统 k-均值和 k-中心点算法局限性

根据图论的理论, 存在两套聚类方案。一种基于互连性(interconnectivity)概念——跨越

两个簇的边的权重之和, 另一种基于接近度(closeness)概念——连接簇 $C_i \sim C_j$ 的顶点的边的平均权重。如图 11-66 和图 11-67 中的例子所示, 单纯采用一个度量准则有时会导致错误的聚类结果。

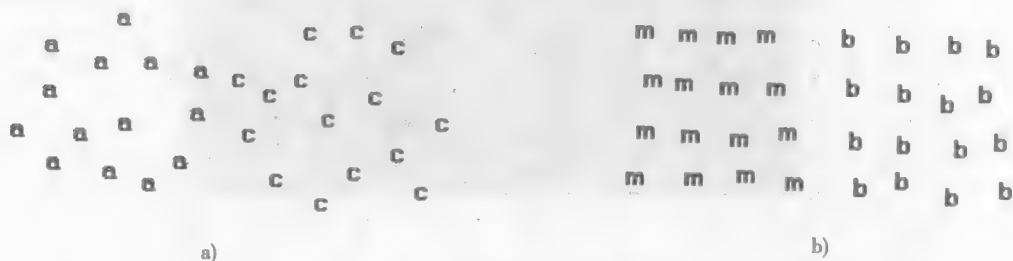


图 11-66 具有不同接近度和互连性度量的两个簇对

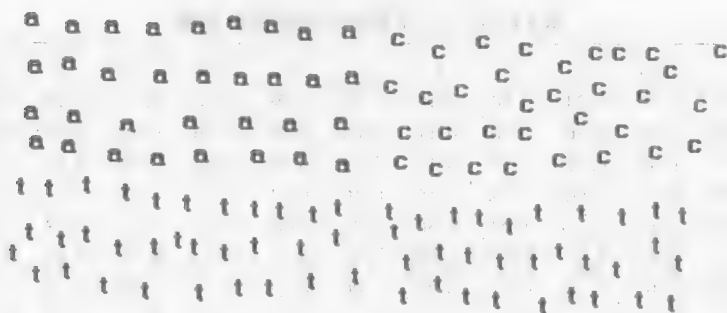


图 11-67 用于解释仅使用互连性度量聚类对象时出现问题的 3 个簇

基于接近度的算法会错误地合并“a”和“c”簇, 因为这两个簇比“m”和“b”簇更接近。

基于互连性的算法会错误合并“a”和“t”簇而不是“a”和“c”簇。现在我们研究基于图论的图划分方法。

11.8.2 平衡图划分——基本原则

给定加权图 $G=(V, E)$, V 表示点集, E 代表边的集合, 加权矩阵为 W 。我们希望利用最小化最大化原则——簇间相似性最小和簇内相似性最大——把数据划分到两个子图 A 和 B 。这是一个可靠的原则, 在统计学、数据挖掘和机器学习领域得到很好证实。节点 u 和 v 之间的相似性度量用边的权重 $W_{u,v}$ 表示。因此, 子图 A 和 B 间的相似性度量是分割粒度(cutsizes)

$$\text{Cut}(A, B) = W(A, B) = \sum_{u \in A, v \in B} W_{u,v}$$

令

$$W(A) = W(A, A), W(B) = W(B, B)$$

现在我们用一个二维数据集来作为例子。如表 11-15 所示, 数据点 1, 2, 3, 4, 9, 10, 11 和 13 属于簇 A , 其余点属于簇 B 。

表 11-15 数据集

点	1	2	3	4	5	6	7	8	9	10	11	12	13	14
X	2	3	3	5	9	9	10	11	1	2	11	12	1	13
Y	2	1	4	3	8	7	10	8	6	7	4	5	7	4

数据点图如图 11-68 所示。表 11-16 给出了相应的相似性度量。

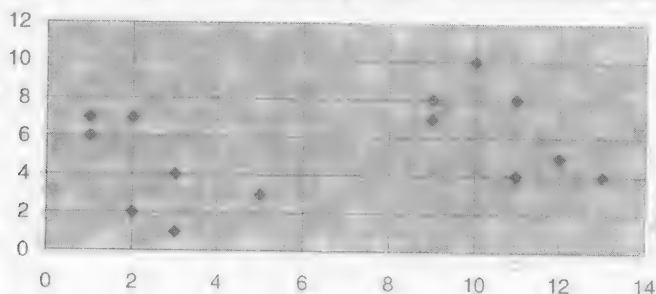


图 11-68 数据点图

表 11-16 14 个数据点的相似性度量

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	1.00	0.41	0.31	0.24	0.10	0.10	0.08	0.08	0.20	0.17	0.10	0.09	0.16	0.08
2	0.41	1.00	0.25	0.26	0.10	0.11	0.08	0.09	0.16	0.14	0.10	0.09	0.14	0.09
3	0.31	0.25	1.00	0.31	0.12	0.13	0.10	0.10	0.26	0.24	0.11	0.10	0.22	0.09
4	0.24	0.26	0.31	1.00	0.14	0.15	0.10	0.11	0.17	0.17	0.14	0.12	0.15	0.11
5	0.10	0.10	0.12	0.14	1.00	0.50	0.31	0.33	0.11	0.12	0.18	0.19	0.11	0.15
6	0.10	0.11	0.13	0.15	0.50	1.00	0.24	0.31	0.11	0.13	0.22	0.22	0.11	0.17
7	0.08	0.08	0.10	0.10	0.31	0.24	1.00	0.31	0.09	0.10	0.14	0.16	0.10	0.13
8	0.08	0.09	0.10	0.11	0.33	0.31	0.31	1.00	0.09	0.10	0.20	0.24	0.09	0.18
9	0.20	0.16	0.26	0.17	0.11	0.11	0.09	0.09	1.00	0.41	0.09	0.08	0.50	0.08
10	0.17	0.14	0.24	0.17	0.12	0.13	0.10	0.10	0.41	1.00	0.10	0.09	0.50	0.08
11	0.10	0.10	0.11	0.14	0.18	0.22	0.14	0.2	0.09	0.10	1.00	0.41	0.09	0.33
12	0.09	0.09	0.10	0.12	0.19	0.22	0.16	0.24	0.08	0.09	0.41	1.00	0.08	0.41
13	0.16	0.14	0.22	0.15	0.11	0.11	0.10	0.09	0.50	0.50	0.09	0.08	1.00	0.07
14	0.08	0.09	0.09	0.11	0.15	0.17	0.13	0.18	0.08	0.08	0.33	0.41	0.07	1.00

点 a 和 b 间的相似性度量采用如下计算公式:

$$S_{a,b} = \frac{1}{1 + \text{Dist}(a, b)}$$

式中 $\text{Dist}(a, b)$ 表示点 a 和 b 间的欧氏距离。

这些相似性度量被看作连接图中节点的权重。例如, $W_{3,5} = 0.12$ 是连接节点 3 和 5 的边的权重。

簇 A (子图 A) 内的相似性或关联是 A 中所有边的权重之和, 等于 $W(A)$ 。节点 u 上的权重 $W_{u,u}$ 包括在 $W(A)$ 中, 这对某些应用来讲非常重要。最小化-最大化聚类原则要求我们使 $\text{Cut}(A, B)$ 最小化, 同时使 $W(A)$ 和 $W(B)$ 最大化。所有这些要求通过以下目标函数可以同时满足 (也就是我们应该把点分配到簇 A 和 B , 使得 Mcut 最小化)。

$$\text{Mcut} = \frac{\text{Cut}(A, B)}{W(A)} + \frac{\text{Cut}(A, B)}{W(B)} \quad (11.12)$$

表 11-15 中带有阴影的点对应于分组 A , 其余的点在分组 B 。我们的问题简化为把元素分为两组, 使得 Mcut 最小。

为了说明该公式的原理, 我们对这些点进行分组并重新安排数据顺序, 如表 11-17 所

示。我们把阴影点(分组 A 中的点)排列在一起。

表 11-17 重新安排的数据点

点	1	2	3	4	5	6	7	8	9	10	11	12	13	14
X	2	3	3	5	1	2	1	9	9	10	11	11	12	13
Y	2	1	4	3	6	7	7	8	7	10	8	4	5	4

对应的相似性矩阵如表 11-18 所示。

表 11-18 表 11-17 中数据的相似性矩阵

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	1.00	0.4	0.3	0.2	0.2	0.2	0.2	0.1	0.1	0.1	0.1	0.1	0.1	0.1
2	0.4	1.00	0.3	0.3	0.2	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
3	0.3	0.3	1.00	0.3	0.3	0.2	0.2	0.1	0.1	0.1	0.1	0.1	0.1	0.1
4	0.2	0.3	0.3	1.00	0.2	0.2	0.2	0.1	0.2	0.1	0.1	0.1	0.1	0.1
5	0.2	0.2	0.3	0.2	1.00	0.4	0.5	0.1	0.1	0.1	0.1	0.1	0.1	0.1
6	0.2	0.1	0.2	0.2	0.4	1.00	0.5	0.1	0.1	0.1	0.1	0.1	0.1	0.1
7	0.2	0.1	0.2	0.2	0.5	0.5	1.00	0.1	0.1	0.1	0.1	0.1	0.1	0.1
8	0.1	0.1	0.1	0.1	0.1	0.1	0.1	1.00	0.5	0.3	0.3	0.2	0.2	0.2
9	0.1	0.1	0.1	0.2	0.1	0.1	0.1	0.5	1.00	0.2	0.3	0.2	0.2	0.2
10	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.3	0.2	1.00	0.3	0.1	0.2	0.1
11	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.3	0.3	0.3	1.00	0.2	0.2	0.2
12	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.2	0.2	0.1	0.2	1.00	0.4	0.3
13	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.2	0.2	0.2	0.2	0.4	1.00	0.4
14	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.2	0.2	0.1	0.2	0.3	0.4	1.00

现在,左上阴影矩阵元素的和就是 $W(A)$ 。 $W(B)$ 是阴影矩阵右下角元素的和。矩阵是对称矩阵, $W(A, B)$ 右上无阴影矩阵或者左下矩阵元素之和。矩阵 W 可表示为:

$$W = \begin{pmatrix} W_A & W_{A,B} \\ W_{A,B} & W_B \end{pmatrix} \quad (11.13)$$

令 D 为一个对角矩阵, 其对角元素为 W 矩阵中相应行或列的元素之和; x 和 y 是被 A, B 共形划分的向量, 即对于表 11-17 中的数据点,

$$x = (1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0)^T$$

而

$$y = (0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1)^T$$

向量 x 中的前 7 个元素为 1, 表示前 7 个数据点(或顶点)属于簇 A 。同样地, 对于向量 y , 它的元素值指明属于簇 B 数据点。注意, x 和 y 是互补的。

对于表 11-15 中的数据点, 向量 x 和 y 分别为:

$$x = (1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0)^T$$

$$y = (0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1)^T$$

由式(11.12)和式(11.13)推导得到:

$$Cut(A, B) = x^T(D - W)x = y^T(D - W)y$$

$$W(A) = x^T W x, \quad W(B) = y^T W y$$

公式 $Cut(A, B)$ 的解释在本书最后附录 B 中给出。对于基于图论的聚类, 目标函数可以被写为(在我们并不知道分组的情况下):

$$Mcut = \frac{\mathbf{x}^T(\mathbf{D} - \mathbf{W})\mathbf{x}}{\mathbf{x}^T\mathbf{W}\mathbf{x}} + \frac{\mathbf{y}^T(\mathbf{D} - \mathbf{W})\mathbf{y}}{\mathbf{y}^T\mathbf{W}\mathbf{y}} \quad (11.14)$$

我们来计算例子的 $Mcut$ 。我们的目标变为寻找使 $Mcut$ 最小的 \mathbf{x} 。

$$W(A) = \mathbf{x}^T\mathbf{W}\mathbf{x} = 17.72, W(B) = \mathbf{y}^T\mathbf{W}\mathbf{y} = 17.68$$

令 $W(T)$ 等于表 11-18 中所有元素之和, $W(T) = 45.26$ 。

$W(A, B)$ 可以通过不同方法得到:

$$W(A, B) = \frac{W(T) - W(A) - W(B)}{2} = 4.93$$

$$Mcut = \frac{4.93}{17.72} + \frac{4.93}{17.68} = 0.558$$

对于这个聚类已知的问题, 这是 $Mcut$ 的最小的可能值。

如果给定的数据点没有被聚类成两组, 我们可以把这个问题转化为一个最优问题, 我们改变向量 \mathbf{x} (或 \mathbf{y}) 使得

$$Mcut = \frac{\mathbf{x}^T(\mathbf{D} - \mathbf{W})\mathbf{x}}{\mathbf{x}^T\mathbf{W}\mathbf{x}} + \frac{\mathbf{y}^T(\mathbf{D} - \mathbf{W})\mathbf{y}}{\mathbf{y}^T\mathbf{W}\mathbf{y}}$$

最小。

11.8.3 k 路划分

到目前为止, 我们一直关注把图分成两个子图。最小-最大切分方法可以扩展到图的平衡 k 路划分。对于 k 路划分, 目标函数为:

$$Mcut_k = \frac{Cut(G_1, \bar{G}_1)}{W(G_1)} + \frac{Cut(G_2, \bar{G}_1)}{W(G_2)} + \dots + \frac{Cut(G_k, \bar{G}_k)}{W(G_k)}$$

11.9 CHAMELEON: 两阶段聚类算法

CHAMELEON[5] 是一种基于图划分的算法。该算法对稀疏图进行操作, 图中节点代表数据项, 加权边表示数据项之间的相似性。数据集的稀疏图表示可以使 CHAMELEON 能够处理大数据集, 并能成功地运用于相似性空间而非度量空间上的数据集。CHAMELEON 采用两阶段算法在数据集中寻找簇。在第一阶段, CHAMELEON 用图划分算法将数据项分成大量的相对较小的子簇。在第二阶段, 它使用凝聚层次聚类算法通过反复结合这些子簇, 最后得到真正的聚类。如图 11-69 所示。

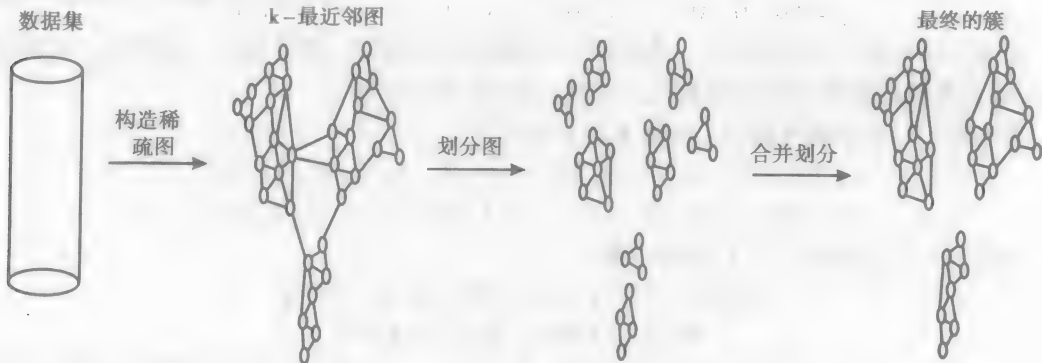


图 11-69 CHAMELEON 的聚类步骤

CHAMELEON 的凝聚层次聚类算法的关键特征是, 它同时考虑互连性性和紧密性特征来确定最相似的子簇对。CHAMELEON 采用新颖的方法, 对簇对的互连性和紧密性度量进行建模, 该方法考虑了簇本身的内部特性。因此, 该算法不依靠用户提供的静态模型, 可以自动适应被合并簇的内部特性。

11.9.1 数据建模

给定一个相似性矩阵, 很多方法都可以用来寻找一种图表示法。事实上, 用图对数据项建模在很多层次聚类算法中是非常常见的。例如, 凝聚层次聚类算法基于单连接、完全连接或者组平均方法对完全图进行操作。

数据项的 CHAMELEON 的稀疏图表示法基于常用的 k -最近邻图方法。 k -最近邻图的每个顶点代表一个数据项, 如果对应于任一节点的数据项是对应于另一节点数据项的 k 个最相似的数据点之一, 则两个顶点之间存在一条边。图 11-70 说明了一个简单数据集的 1-最近邻、2-最近邻和 3-最近邻图。注意, 由于 CHAMELEON 在稀疏图上进行操作, 因此, 每个簇只不过是表示数据集的原始稀疏图的一个子图。

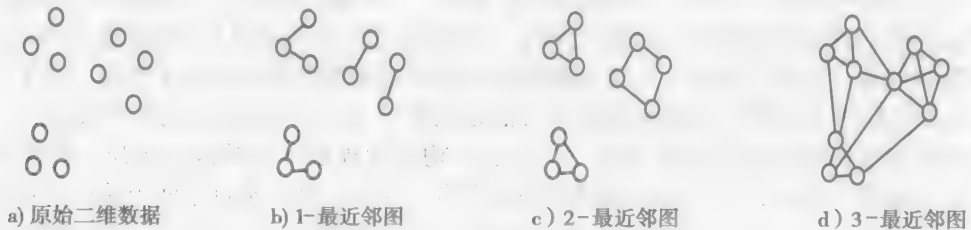


图 11-70 简单数据集的图

采用 k -最近邻图 G_k 表示数据具有一些优点。首先, 离得很远的数据点在 G_k 中是完全不连通的。其次, G_k 动态地捕捉邻域概念。一个数据点所在区域的密度决定了该点的邻域半径。在稠密区域, 邻域被定义得比较窄; 而在稀疏区域, 邻域被定义得比较宽。在 DBSCAN 定义的模型中指定了一个全局邻域密度, 与之相比, G_k 捕获更自然的邻域。第三, 区域密度作为边的权重。稠密区域 G_k 的边权重(边权重表示相似性)会较大, 而稀疏区域的边权重较小。因此, 一个图的最小二分割集表示稀疏区域分界层。最后, 与其他一些在完整图上进行操作的算法(包括图划分和划分精炼算法)相比, G_k 具有计算上的优势。

11.9.2 簇相似性建模

11.3 节讨论了凝聚方案的局限性, CHAMELEON 通过同时考虑每对簇 C_i 和 C_j 的相对互连度 $RI(C_i, C_j)$ 和相对紧密度 $RC(C_i, C_j)$ 来确定它们的相似性。CHAMELEON 的层次聚类算法选择合并相对互连度 $RI(C_i, C_j)$ 和相对紧密度 $RC(C_i, C_j)$ 都高的簇对。也就是说, 它选择合并这样的簇, 对于簇的内部互连度和紧密度而言, 它们是高度互连和紧密的。通过基于这两个准则选择簇, CHAMELEON 克服了那些仅考虑绝对互连度或绝对紧密度算法的局限性。例如, 图 11-66 和图 11-67 所示的例子和在 11.9.1 节所讨论的例子中, CHAMELEON 会选择合并正确的簇对。在本节的剩余部分, 我们将讲述如何计算一对簇之间的相对互连度和相对紧密度。

相对互连度: 一对簇 C_i 和 C_j 之间的相对互连度被定义为, 用簇 C_i 和 C_j 的内部互连度规格化的簇 C_i 和 C_j 之间的绝对互连度。簇 C_i 和 C_j 之间的绝对互连度是连接簇 C_i 和 C_j

的顶点的所有边的权重之和,其本质是同时包含 C_i 和 C_j 的簇的边割(edge-cut),以便将这个簇分为 C_i 和 C_j 。我们用 $EC(C_i, C_j)$ 表示。簇 C_i 的内部互连度可以通过它的最小二分割 $EC(C_i)$ 的大小(即将图划分成两个大致相等的部分的边的加权之和)表示。近来在图划分技术上的进步,使得寻找图的最小二分割变得非常高效。簇 C_i 和 C_j 间的相对互连度由下式给出:

$$RI(C_i, C_j) = \frac{|EC(C_i, C_j)|}{\frac{|EC(C_i)| + |EC(C_j)|}{2}} \quad (11.15)$$

上式用两个簇的平均内部互连度来规格化绝对互连度。这里, $|EC(C_i, C_j)|$ 表示绝对互连度,可以表示为:

$|EC(C_i, C_j)|$ = 跨越两个簇的所有边的权重和

$EC(C_i)$ 表示内部互连度,可以写为:

$EC(C_i)$ = 簇 C_i 内所有边的权重和

相对紧密度: 簇 C_i 和 C_j 之间的相对紧密度(如图 11-71 所示)被定义为用簇 C_i 和 C_j 的内部紧密度规格化的簇 C_i 和 C_j 之间的绝对紧密度。一对簇之间的绝对紧密度可以通过多种不同方法获得。许多已有的方案从簇 C_i 和 C_j 中的所有点(或代表点)中寻找最近的点对。由于仅依靠一对点,这种方案的一个重大缺陷是对离群点和噪声的容忍程度不够。由于这个原因,CHAMELEON 度量两个簇的紧密度的方法是取簇 C_i 和 C_j 间连接点的平均相似性。由于这些连接是通过 k -最近邻图确定的,它们的平均强度提供了反映簇分界层中数据项间亲和力的良好度量。同时,可以容忍离群数据和噪声。注意,这个来自于两个簇中的点间的平均相似性等于连接 C_i 和 C_j 之间的边的平均权重。在 CHAMELEON 算法中,相对紧密度由如下公式获得:

$$RC(C_i, C_j) = \frac{SEC(C_i, C_j)}{\frac{|C_i|}{|C_i| + |C_j|} SEC(C_i) + \frac{|C_j|}{|C_i| + |C_j|} SEC(C_j)} \quad (11.16)$$

式中,绝对紧密度 $SEC(C_i, C_j)$ = 类 C_i 和 C_j 的连接边的平均权重, $SEC(C_i)$ 是内部紧密度。

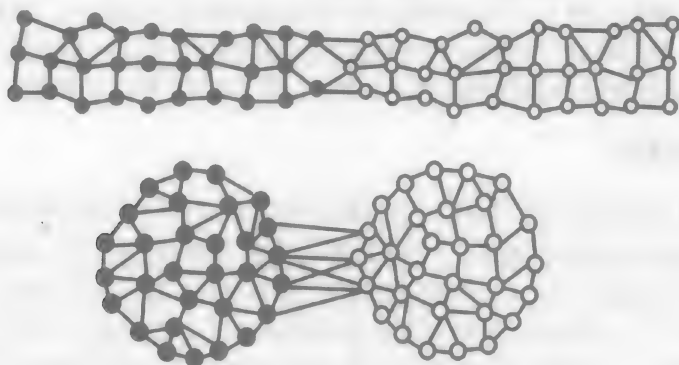


图 11-71 第一对簇更“紧密”

11.9.3 CHAMELEON 的两个阶段

CHAMELEON 算法包含两个不同的阶段。第一个阶段的目的是将所有数据项聚集成大量的子簇,这些子簇包含足够数量的数据项,以便进行动态建模。第二个阶段的目的是通过采

用动态建模框架以层次方式来合并子簇, 最终得到真正的聚类。下面我们将介绍 CHAMELEON 的两个阶段所采用的算法。

阶段 I: 发现初始子簇。CHAMELEON 采用图划分算法将数据集 k -最近邻图划分为大量的子簇, 使得切边最小化, 也就是跨越两个子簇的边的权重和最小。既然 k -最近邻图中的每条边表示点间的相似性, 最小化割边的划分就相当于最小化了跨越结果划分的点之间的关系(亲和力)。一个潜在的假设是簇内的连接比簇间的连接强度更高、更丰富。因此, 在每个划分内的数据项间的联系更紧密。

阶段 II: 采用动态框架合并子簇。一旦在第一阶段采用基于划分的算法得到细粒度的聚类结果, CHAMELEON 就采用凝聚层次聚类来合并这些小的子簇。正如在 11.2 节所讨论的, 凝聚层次聚类算法的关键步骤是寻找最相似的子簇对。CHAMELEON 的凝聚层次聚类算法采用前面所述的动态建模框架, 同时考虑簇之间的相对互连度和相对紧密度来选择最相似的簇对。同时考虑以上两种度量, 可以提出很多凝聚层次聚类算法。CHAMELEON 算法实现了两种不同的方案。第一种方案是仅合并那些相对互连性和相对紧密度都高于用户指定的阈值 T_{RI} 和 T_{RC} 的簇对。在这种方法中, CHAMELEON 访问每个簇 C_i , 并核查其邻接簇 C_j 是否满足下面两个条件:

$$RI(C_i, C_j) > T_{RI} \text{ 且 } RC(C_i, C_j) > T_{RC} \quad (11.17)$$

如果不止一个相邻的簇满足上述条件, CHAMELEON 选择合并与 C_i 联系最紧密的簇, 即选择与簇 C_i 绝对互连性最高的簇 C_j 。一旦每个簇都得到了与它的一个相邻的近邻合并的机会, 就执行选择的合并, 并重复整个过程。注意, 这个算法与传统层次聚类算法不同, 在一次迭代中它允许合并多对簇。参数 T_{RI} 和 T_{RC} 用来控制期望簇的特性。特别是, 参数 T_{RI} 允许我们控制簇中数据项的互连程度的可变性。参数 T_{RC} 允许我们控制特定簇内数据项间相似程度的同一性。依据参数 T_{RI} 和 T_{RC} 的设置, 由于没有邻近的簇满足式(11.17)给出的条件, 因此 CHAMELEON 的合并算法可能无法继续下去。这时, 我们可以选择终止算法并输出当前聚类作为解, 或者尝试通过按不同比例松弛两个参数, 继续合并另外的簇对。

CHAMELEON 算法采取的第二种方案是用一个函数来结合相对互连性和相对紧密性, 然后选择合并最大化该函数的簇对。既然我们的目的是为了合并相对互连度和相对紧密度都高的簇对, 定义这样函数的自然的一种方法是取它们的乘积。也就是说, 选择合并簇对 C_i 和 C_j , 使得 $RI(C_i, C_j) * RC(C_i, C_j)^\alpha$ 最大。在这个公式中, 两个参数重要性相同。然而, 我们常常更偏好使得两个度量中的某一个更高的簇。因为这个原因, CHAMELEON 算法选择这样的簇对, 它们最大化

$$RI(C_i, C_j) * RC(C_i, C_j)^\alpha \quad (11.18)$$

式中, α 是一个用户指定参数。如果 $\alpha > 1$, 则 CHAMELEON 给相对紧密度赋予更高的重要性。如果 $\alpha < 1$, 则相对互连性具有更高的重要性。在图 11-72 所示的实验结果中, CHAMELEON 算法的作者采用第二种方法, 我们可以很容易地为层次聚类生成完整的树状图。

11.9.4 用例子说明 CHAMELEON 算法

重新考虑图划分例子中给出的数据, 数据



图 11-72 实验结果

重新显示在图 11-73 中。

Points	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1	1.00	0.41	0.31	0.24	0.10	0.10	0.08	0.08	0.20	0.17	0.18	0.09	0.10	0.08			
2	0.41	1.00	0.25	0.26	0.10	0.11	0.08	0.09	0.16	0.14	0.14	0.09	0.10	0.09			
3	0.31	0.25	1.00	0.31	0.12	0.13	0.10	0.10	0.26	0.24	0.22	0.10	0.11	0.09			
4	0.24	0.26	0.31	1.00	0.14	0.15	0.10	0.11	0.17	0.17	0.15	0.12	0.14	0.11			
5	0.10	0.10	0.12	0.14	1.00	0.50	0.31	0.33	0.11	0.12	0.11	0.19	0.18	0.15			
6	0.10	0.11	0.13	0.15	0.50	1.00	0.24	0.31	0.11	0.13	0.11	0.22	0.22	0.17			
7	0.08	0.09	0.10	0.11	0.31	0.24	1.00	0.31	0.09	0.10	0.10	0.16	0.14	0.13			
8	0.08	0.09	0.10	0.11	0.31	0.24	0.31	1.00	0.09	0.10	0.09	0.24	0.20	0.18			
9	0.20	0.16	0.26	0.17	0.11	0.11	0.09	0.09	1.00	0.41	0.50	0.08	0.09	0.08			
10	0.17	0.14	0.24	0.17	0.12	0.13	0.10	0.10	0.41	1.00	0.50	0.09	0.10	0.08			
11	0.18	0.14	0.22	0.15	0.19	0.11	0.10	0.09	0.50	0.50	1.00	0.08	0.09	0.07			
12	0.09	0.09	0.10	0.12	0.19	0.22	0.16	0.24	0.08	0.09	0.08	1.00	0.41	0.41			
13	0.10	0.10	0.11	0.14	0.18	0.22	0.14	0.20	0.09	0.10	0.09	0.41	1.00	0.33			
14	0.08	0.09	0.09	0.11	0.15	0.17	0.13	0.18	0.08	0.08	0.07	0.41	0.33	1.00			

图 11-73 相似性度量

在 CHAMELEON 聚类算法的第一阶段，我们采用图划分方法得到细粒度聚类。对于所考虑数据，我们得到四个簇，即

簇 1: $C_1 = \{(2, 2), (3, 1), (3, 4), (5, 3)\}$ ，这些点的索引集为 $\{1, 2, 3, 4\}$ 。

簇 2: $C_2 = \{(9, 8), (9, 7), (10, 10), (11, 8)\}$ ，这些点的索引集为 $\{5, 6, 7, 8\}$ 。

簇 3: $C_3 = \{(1, 6), (2, 7), (1, 7)\}$ ，这些点的索引集为 $\{9, 10, 13\}$ 。

簇 4: $C_4 = \{(11, 4), (12, 5), (13, 4)\}$ ，这些点的索引集为 $\{11, 12, 14\}$ 。

为了便于说明，我们对数据进行重新排序，使得同一组内的点是连续的。图 11-73 显示了重新排序的点及其相似性度量。

在算法的第二阶段，基于相对互连性(RI)和相对紧密度度量，我们寻找可以被合并的簇对。为了得到 RI ，我们需要每个簇对的 $|EC(C_i, C_j)|$ ($=$ 所有跨越两个簇的边的权重之和)和 $EC(C_i)$ (表示 C_i 的内部互连性)。 $RI(C_i, C_j)$ 由下式给出：

$$RI(C_i, C_j) = \frac{|EC(C_i, C_j)|}{\frac{1}{2}(|EC(C_i)| + |EC(C_j)|)} \quad (11.19)$$

表 11-19 显示了由图 11-73 中的相似性度量计算得到的 $|EC(C_i, C_j)|$ 。表 11-19 中的每个元素对应于图 11-73 中阴影区域和非阴影区域不同块中所有元素和。

表 11-19 簇对的互连性(非对角线)和内部互连性(对角线)度量

	1	2	3	4
1	7.567	1.691	2.162	1.226
2	1.691	8.001	1.260	2.175
3	2.162	1.260	5.828	0.758
4	1.226	2.175	0.758	5.323

由表 11-19，我们采用式(11.19)计算每个簇对的相对互连性度量。结果显示在表 11-20 中。

表 11-20 相对互连性度量

	1	2	3	4
1	—	0.22	0.32	0.19
2	0.22	—	0.18	0.33
3	0.32	0.18	—	0.14
4	0.19	0.33	0.14	—

利用表 11-20 和已知的每个簇的基(簇中元素的数目), 我们按下列公式计算相对紧密度, 表 11-21 给出了计算结果。

$$RC(C_i, C_j) = \frac{\overline{SEC}(C_i, C_j)}{\frac{|C_i|}{|C_i| + |C_j|} \overline{SEC}(C_i) + \frac{|C_j|}{|C_i| + |C_j|} \overline{SEC}(C_j)} \quad (11.20)$$

表 11-21 相对紧密性度量

	1	2	3	4
1	—	0.22	0.33	0.19
2	0.22	—	0.19	0.34
3	0.33	0.19	—	0.14
4	0.19	0.34	0.14	—

接下来, 我们结合这两个度量来决定要合并的簇。为达到这个目的, 通过把表 11-20 和表 11-21 中的元素相乘得到表 11-22。

表 11-22 互连性和紧密性度量积 ($RI^* RC$)

	1	2	3	4
1	—	0.05	0.11	0.037
2	0.05	—	0.03	0.11
3	0.11	0.03	—	0.018
4	0.037	0.11	0.018	—

我们发现簇对(1, 3)和(2, 4)使得 $RI^* RC$ 最大化。因此每个簇对可以合并在一起。

为了画出树状图, 我们需要计算簇之间的距离, 这就要从平均紧密度度量推导 $EC(C_1, C_3) = 2.162$, 这是簇 C_1 与 C_3 内 12 个点的相似性度量和。因此, 平均值 $\overline{EC}(C_1, C_3) = \frac{2.162}{12} = 0.18$ 。我们可以利用关系 $S_{ij} = \frac{1}{1 + d_{ij}}$ 或者 $d_{ij} = \frac{1}{S_{ij}} - 1$ 将它转换为距

离度量。因此, $d_{13} = \frac{1}{0.18} - 1 = 4.55$ 。类似地, $d_{24} = 4.52$ 。

现在我们只剩两个簇, 分别用 C_{13} 和 C_{14} 表示。

$EC(C_{13}, C_{14}) = 27.3$, $\overline{EC}(C_{13}, C_{14}) = \frac{4.93}{49} = 0.1001$, $d_{13,24} = \frac{1}{0.1001} - 1 = 9$

结果树状图显示在图 11-74 中。

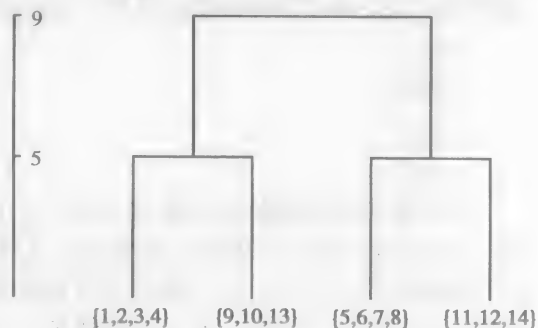


图 11-74 树状图

11.10 COBWEB 概念聚类算法

COBWEB 算法是由机器学习研究者在 20 世纪 80 年代提出的,用于在对象-属性数据集中对对象进行聚类。它产生一棵聚类树状图,称为分类树(classification tree)。该树用概率描述来刻画每个簇。这个算法基于所谓的分类效用(Category Utility, CU)函数来度量聚类质量。如果把一组对象划分到 m 个簇,则这个划分的 CU 表示为:

$$CU = \frac{\sum_{k=1}^m P(C_k) \left[\sum_i \sum_j P(A_i = V_{ij} | C_k)^2 - \sum_i \sum_j P(A_i = V_{ij})^2 \right]}{m} \quad (11.21)$$

式中, A_i 表示第 i 个属性, V_{ij} 是属性 A_i 的第 j 个属性值。 C_k 是第 k 个簇, m 代表簇数目。 $P(C_k)$ 是簇 C_k 中的数据所占的比例。

我们用下面的例子数据来解释这些概念。

这里,属性 A_1 表示颜色, A_2 是核, A_3 是尾。
 A_1 取两个值 V_{11} = 白色或者 V_{12} = 黑色。类似地,
 $V_{21} = 1$, $V_{22} = 2$, $V_{23} = 3$ 是属性 A_2 的可能取值。

式(11.21)看起来很令人费解,但是一旦你领悟了它的意思和重要性便容易理解了。公式中所包含的各个项的标准解释如表 11-23 所示。

表 11-23 数据集

颜色	核	尾
白色	1	1
白色	2	2
黑色	2	2
黑色	3	1

给定簇 C_k 中的一个对象,如果我们根据出现的概率估计它的属性值,那么可以正确猜测的属性值数量为:

$$\sum_i \sum_j P(A_i = V_{ij} | C_k)^2 \quad (11.22)$$

给定一个其簇未知的对象,如果我们根据出现的概率估计它的属性值,那么我们期望可以正确猜测的属性值数量由这个公式不能得到。 $P(C_k)$ 被合并到 CU 函数,给每个簇分配适当的权重。

最后, m 被放置在分母位置防止过拟合。

在本章的结尾处,我们讨论该公式的含义。

11.10.1 COBWEB 算法

COBWEB 算法通过把对象逐个插入到分类树中来增量地构造分类树。当要把一个对象插入到分类树时,COBWEB 算法从根节点开始自顶向下地遍历树。在访问每个节点时,COBWEB 算法考虑四种可能的操作,并选择其中一种使 CU 函数值最大的操作:

- 插入
- 创建
- 合并
- 分裂

插入操作表示将新的对象插入到一个已存在的子节点。COBWEB 算法估算将新对象插入已存在的子节点后的 CU 函数值,并选择取值最大的节点。COBWEB 算法也考虑为新对象创建新的子节点。为了避免簇对数据集输入顺序的依赖,COBWEB 算法考虑合并 CU 函数值最高的两个子节点和分裂取值最大的子节点。图 11-75 和 11-76 表示了合并和分裂操作。

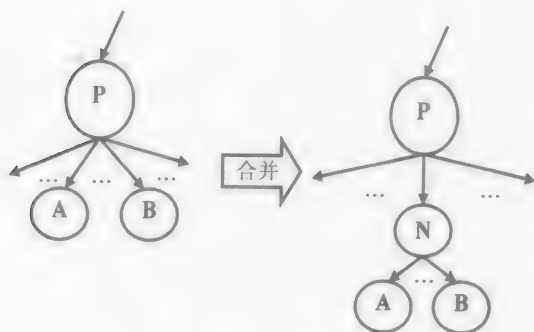


图 11-75 合并

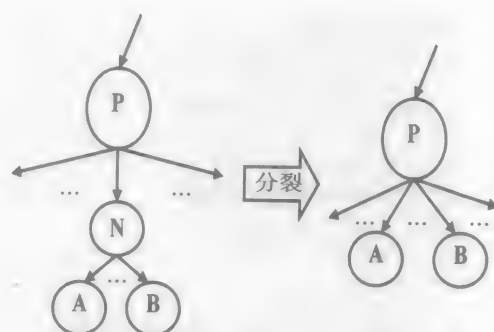


图 11-76 分裂

完整的 COBWEB 算法在下文中给出,之后我们用一个例子来说明 COBWEB 算法,最后讨论分类效用函数。

COBWEB 算法

输入:

概念层次的当前节点 N 。

一个未分类(属性-值)实例 I 。

结果:

用来分类实例的概念层次

顶层调用: Cobweb (Top-node, I)

变量: C, P, Q 和 R 是层次结构中的节点

U, V, W 和 X 是聚类(划分)得分

方法:

Cobweb (N, I)

If N 为末端节点

Then Create-new-terminals (N, I);

Incorporate (N, I);

Else Incorporate (N, I);

For 节点 N 的每个子节点 C do

计算将 I 放入 C 的得分;

令 P 是具有最高得分 W 的节点;

令 Q 是具有次高分的节点;

令 X 是将 I 放入新节点 R 的得分;

令 Y 是将 P 和 Q 合并为一个节点的得分;

令 Z 是将 P 划分到其子节点的得分;

If W 是最高得分 Then

Cobweb (P, I) (将 I 放入类别 P);

Else if X 是最高得分 Then

使用 I 的值初始化 R 的概率;

(将 I 放入新类别 R);

Else if Y 是最高得分 Then

令 O 是 Merge (P, Q, N);

Cobweb (O, I);

Else if Z 是最高得分 Then

Split (P, N);

Cobweb (N, I).

辅助 COBWEB 操作:

变量: N, O, P 和 R 是层次结构中的节点;

I 是一个未分类的实例;

A 是标称属性;

V 是一个属性值;

Incorporate (N, I)

```

更新类别 N 的概率;
For 实例 I 中的每个属性 A do
    For A 的每个值 V do
        更新给定类别 N 中 V 的概率;
Create-new-terminals(N, I)
    创建节点 N 的一个新子节点 M;
    用 N 中的概率初始化 M 的概率;
    创建节点 N 的一个新子节点 O;
    用 I 的值初始化 O 的概率;
Merge(P, R, N)
    令 O 为 N 的新子节点;
    将 O 的概率设置为 P 和 R 的概率的平均值;
    从节点 N 移出子节点 P 和 R;
    将 P 和 R 添加为节点 O 的子节点;
    Return O;
Split(P, N)
    删除节点 N 的子节点 P;
    将 P 的子节点提升为 N 的子节点;

```

11.10.2 COBWEB: 一个简单例子

考虑表 11-24 中的数据。让我们遵循调用算法时事件发生顺序。这里属性 A_1 代表颜色, A_2 表示核, A_3 是尾。 A_1 可取值 V_{11} (白色) 或 V_{12} (黑色)。 A_2 的值域为 $V_{21} = 1$, $V_{22} = 2$ 和 $V_{23} = 3$ 。同样地, A_3 可取的值为 $V_{31} = 1$ 和 $V_{32} = 2$ 。

表 11-24 示例数据

实例标签	属 性		
	颜色	核	尾
a	白色	1	1
b	白色	2	2
c	黑色	2	2
d	黑色	3	1

现在, 我们取第一个标签为“a”的实例, 其属性颜色 = 白色, 核 = 1, 尾 = 1。创建一个根节点。图 11-77 表示根节点和位于其下的数据的全部概要。

		C_1	$P(C_1) = 1$
	属性	值	P
根	尾	1	1.0
		2	0.0
	颜色	白色	1.0
		黑色	0.0
	核	1	1.0
		2	0.0
		3	0.0

图 11-77 第一次迭代后根节点和它的内容

随着我们增加更多数据, 根节点将包含叶节点数据的全部统计信息。

将第二个数据插入树中时, 我们将它看作新的实例 I , 并再次调用 $COBWEB(Root, I)$ 。由于当前的根节点是一个叶节点, 我们调用函数 $Create-new-Terminal(Root, I)$ 产生两个新的叶节点; 分别把根节点和实例 I 内容复制到两个叶节点。算法调用函数 $INCORPO-$

$RATE(Root, I)$ 来更新根节点的概率,如图 11-78 所示。

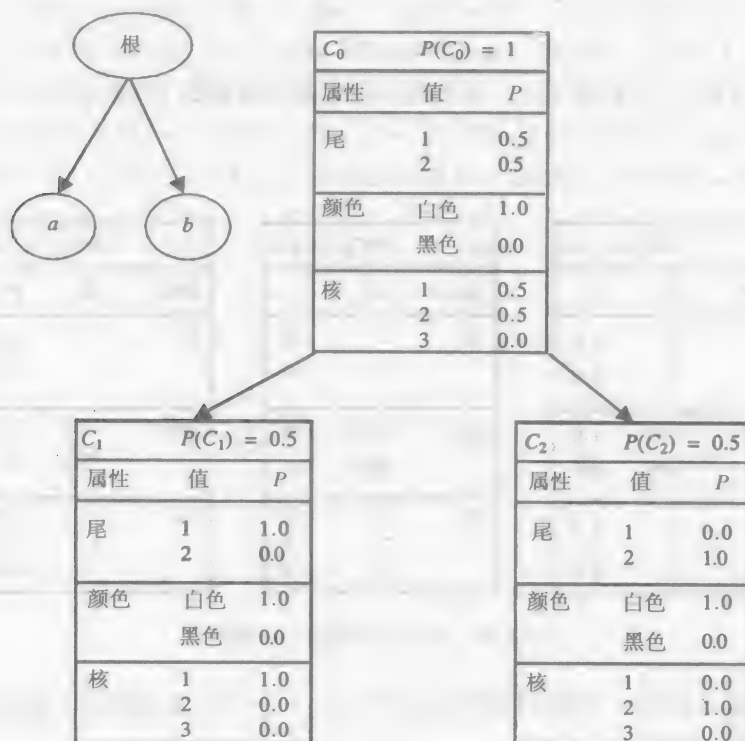


图 11-78 插入第二个数据后的树

现在我们把标记为“c”的数据作为实例 I 并调用 $COBWEB(Root, I)$ 。因为根节点不是叶节点,算法调用 $INCORPORATE(Root, I)$ 并更新根节点的概率值。根节点的概率如图 11-79 所示。

随后,算法通过暂时把 I 放置在根节点的子节点 C_1 和 C_2 , 分别计算 CU 值。当放置在 C_1 时,条件概率暂时改变,如图 11-80 所示。除了 $P(C_2)$, C_2 内的条件概率保持不变。

C_0	$P(C_0) = 1.0$	
属性	值	P
尾	1	1/3
	2	2/3
颜色	白色	2/3
	黑色	1/3
核	1	1/3
	2	2/3
	3	0.0

C_1	$P(C_1) = 2/3$	
属性	值	P
尾	1	0.5
	2	0.5
颜色	白色	1.0
	黑色	0.0
核	1	0.5
	2	0.5
	3	0.0

C_2	$P(C_2) = 1/3$	
属性	值	P
尾	1	0.0
	2	1.0
颜色	白色	1.0
	黑色	0.0
核	1	0.0
	2	1.0
	3	0.0

C_0	$P(C_0) = 1.0$	
属性	值	P
尾	1	1/3
	2	2/3
颜色	白色	2/3
	黑色	1/3
核	1	1/3
	2	2/3
	3	0.0

图 11-79 更新后的根节点

图 11-80 叶节点和根节点的概率

然后, 我们计算 CU_1 :

$$CU_1 = [P(C_1) * (0.5^2 + 0.5^2 + 1^2 + 0^2 + 0.5^2 + 0.5^2 + 0^2) + P(C_2) * (0^2 + 1^2 + 1^2 + 0^2 + 0^2 + 1^2 + 0^2) - (0.33^2 + 0.67^2 + 0.67^2 + 0.33^2 + 0.33^2 + 0.67^2 + 0^2)] / 2$$

现在把 I 放置在 C_2 并计算 CU_2 , 条件概率和无条件概率如下(参见图 11-81):

$$CU_2 = [P(C_1) * (1^2 + 0^2 + 1^2 + 0^2 + 1^2 + 0^2 + 0^2) + P(C_2) * (0^2 + 1^2 + 0.5^2 + 0.5^2 + 0^2 + 1^2 + 0^2) - (0.33^2 + 0.67^2 + 0.67^2 + 0.33^2 + 0.33^2 + 0.67^2 + 0^2)] / 2$$

C_1 $P(C_1) = 1/3$			C_2 $P(C_2) = 2/3$			C_0 $P(C_0) = 1.0$		
属性	值	P	属性	值	P	属性	值	P
尾	1	1.0	尾	1	0.0	尾	1	1/3
	2	0.0		2	1.0		2	2/3
颜色	白色	1.0	颜色	白色	0.5	颜色	白色	2/3
	黑色	0.0		黑色	0.5		黑色	1/3
核	1	1.0	核	1	0.0	核	1	1/3
	2	0.0		2	1.0		2	2/3
	3	0.0		3	0.0		3	0.0

图 11-81 叶节点和根节点的概率

下一步, 放置 I 作为一个单独的节点并计算 CU_3 , 三个叶节点的条件概率为(如图 11-82 所示):

$$CU_3 = [P(C_1) * (1^2 + 0^2 + 1^2 + 0^2 + 1^2 + 0^2 + 0^2) + P(C_2) * (0^2 + 1^2 + 1^2 + 0^2 + 0^2 + 1^2 + 0^2) + P(C_3) * (0^2 + 1^2 + 0^2 + 1^2 + 0^2 + 1^2 + 0^2) - (0.33^2 + 0.67^2 + 0.67^2 + 0.33^2 + 0.33^2 + 0.67^2 + 0^2)] / 3$$

C_1 $P(C_1) = 1/3$			C_2 $P(C_2) = 1/3$			C_3 $P(C_3) = 1/3$		
属性	值	P	属性	值	P	属性	值	P
尾	1	1.0	尾	1	0.0	尾	1	0.0
	2	0.0		2	1.0		2	1.0
颜色	白色	1.0	颜色	白色	1.0	颜色	白色	0.0
	黑色	0.0		黑色	0.0		黑色	1.0
核	1	1.0	核	1	0.0	核	1	0.0
	2	0.0		2	1.0		2	1.0
	3	0.0		3	0.0		3	0.0

图 11-82 叶节点的概率

由于 CU_2 最大, 我们沿 C_2 放置 I 。为此, 调用 $COBWEB(C_2, I)$ 。(建议读者再复习一下算法。)结果树如图 11-83 和 11-84 所示。

当前节点 C_2 包含子节点 C_3 和 C_4 的汇总统计信息。现在, 我们取最后一个数据集, 并标记为实例 I 。调用 $COBWEB(Root, I)$ 。由于当前根节点不是叶节点, 调用 $INCORPORATE(Root, I)$ 并更新根节点概率值。根节点概率见图 11-85。

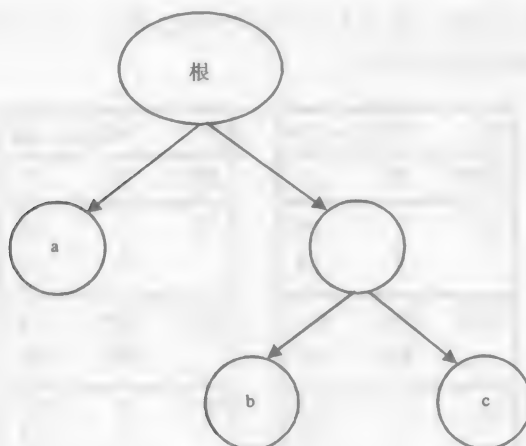


图 11-83 插入第三个数据后的树

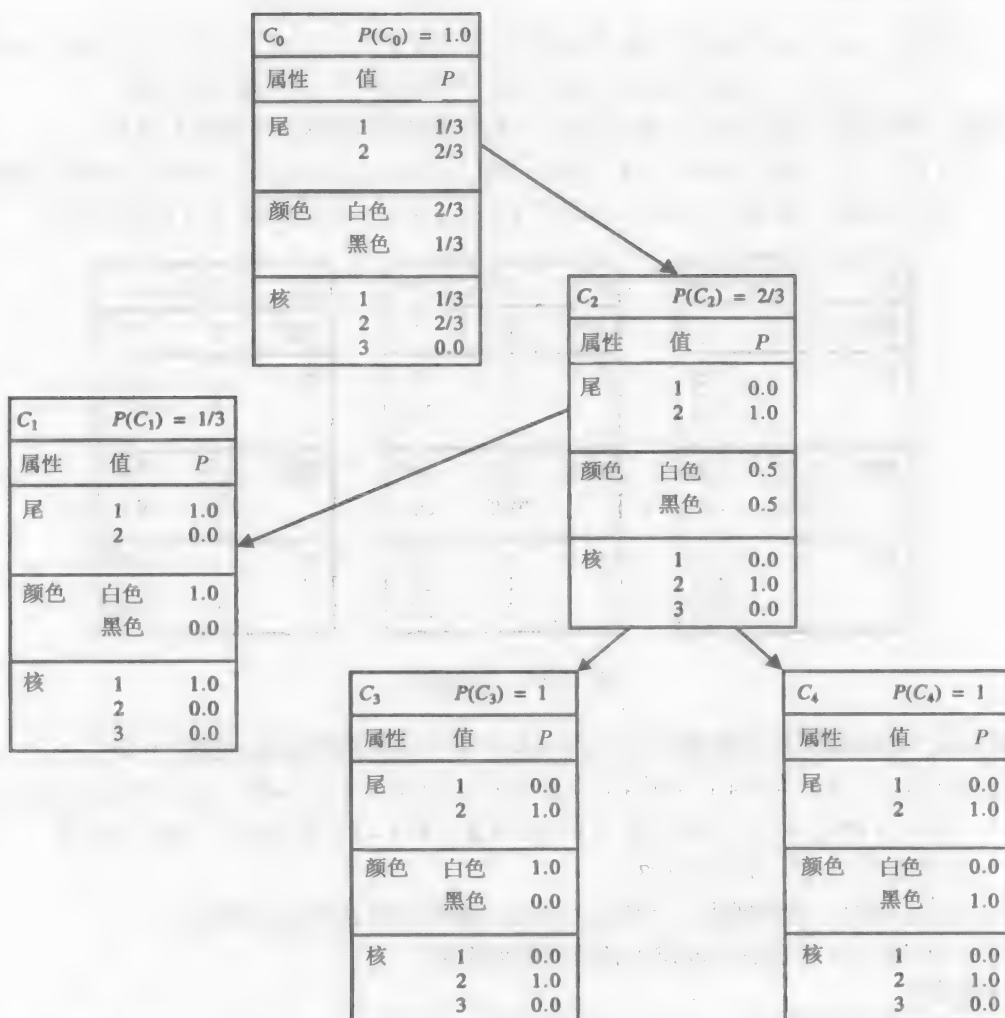


图 11-84 插入第三个数据后的节点概率

算法计算把 I 暂时放在根的子节点 C_1 时的 CU 值。此时, C_1 的概率暂时变为如图 11-86 所示。除了 $P(C_2)$, C_2 的概率保持不变。

$C_0 \quad P(C_0) = 1.0$		
属性	值	P
尾	1	0.5
	2	0.5
颜色	白色	0.5
	黑色	0.5
核	1	0.25
	2	0.5
	3	0.25

图 11-85 根节点概率

$C_1 \quad P(C_1) = 2/4$		
属性	值	P
尾	1	1.0
	2	0.0
颜色	白色	0.5
	黑色	0.5
核	1	0.5
	2	0.0
	3	0.5

$C_2 \quad P(C_2) = 2/4$		
属性	值	P
尾	1	0.0
	2	1.0
颜色	白色	0.5
	黑色	0.5
核	1	0.0
	2	1.0
	3	0.0

图 11-86 叶节点概率

$C_0 \quad P(C_0) = 1$		
属性	值	P
尾	1	0.5
	2	0.5
颜色	白色	0.5
	黑色	0.5
核	1	0.25
	2	0.5
	3	0.25

$$CU_1 = [P(C_1) * (1^2 + 0^2 + 0.5^2 + 0.5^2 + 0.5^2 + 0^2 + 0.5^2) + P(C_2) * (0^2 + 1^2 + 0.5^2 + 0.5^2 + 0^2 + 1^2 + 0^2) - (0.5^2 + 0.5^2 + 0.5^2 + 0.5^2 + 0.25^2 + 0.5^2 + 0.25^2)]/2$$

现在, 我们将 I 放到节点 C_2 并计算 C_2 。条件和非条件概率为(见图 11-87):

$$CU_2 = [P(C_1) * (1^2 + 0^2 + 1^2 + 0^2 + 1^2 + 0^2 + 0.0^2) + P(C_2) * (0.33^2 + 0.67^2 + 0.33^2 + 0.67^2 + 0^2 + 0.67^2 + 0.33^2) - (0.5^2 + 0.5^2 + 0.5^2 + 0.5^2 + 0.25^2 + 0.5^2 + 0.25^2)]/2$$

$C_1 \quad P(C_1) = 2/4$		
属性	值	P
尾	1	1.0
	2	0.0
颜色	白色	1.0
	黑色	0.0
核	1	1.0
	2	0.0
	3	0.0

$C_2 \quad P(C_2) = 3/4$		
属性	值	P
尾	1	1/3
	2	2/3
颜色	白色	1/3
	黑色	2/3
核	1	0.0
	2	2/3
	3	1/3

$C_0 \quad P(C_0) = 1$		
属性	值	P
尾	1	0.5
	2	0.5
颜色	白色	0.5
	黑色	0.5
核	1	0.25
	2	0.5
	3	0.25

图 11-87 节点概率

接下来, 我们将 I 作为单独的节点 C_5 。这三个子节点的概率为(见图 11-88):

$$CU_5 = [P(C_1) * (1^2 + 0^2 + 1^2 + 0^2 + 1^2 + 0^2 + 0^2) + P(C_2) * (1^2 + 0^2 + 0.5^2 + 0.5^2 + 0^2 + 1^2 + 0^2) + P(C_5) * (1^2 + 0^2 + 0^2 + 1^2 + 0^2 + 0^2 + 1^2) - (0.5^2 + 0.5^2 + 0.5^2 + 0.5^2 + 0.25^2 + 0.5^2 + 0.25^2)]/3$$

由于 CU_5 最大, 我们创建一个独立的节点。最后的簇如图 11-89 所示。

图 11-90 显示了在最后生成树中各节点的概率值。

分类效用

现在我们考虑用来度量划分的整体质量的分类效用是如何计算的。分类效用的定义相当艰深。

$C_1 \quad P(C_1) = 1/4$		
属性	值	P
尾	1	1.0
	2	0.0
颜色	白色	1.0
	黑色	0.0
核	1	1.0
	2	0.0
	3	0.0

$C_2 \quad P(C_2) = 2/4$		
属性	值	P
尾	1	1.0
	2	0.0
颜色	白色	0.5
	黑色	0.5
核	1	0.0
	2	1.0
	3	0.0

$C_3 \quad P(C_3) = 1/4$		
属性	值	P
尾	1	1.0
	2	1.0
颜色	白色	0.0
	黑色	1.0
核	1	0.0
	2	0.0
	3	1.0

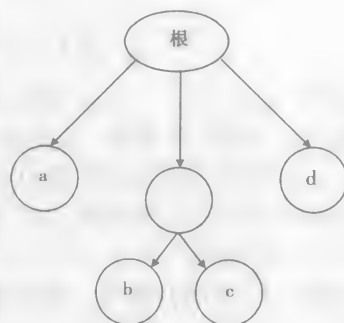


图 11-88 叶节点概率

图 11-89 插入第四个数据后的树

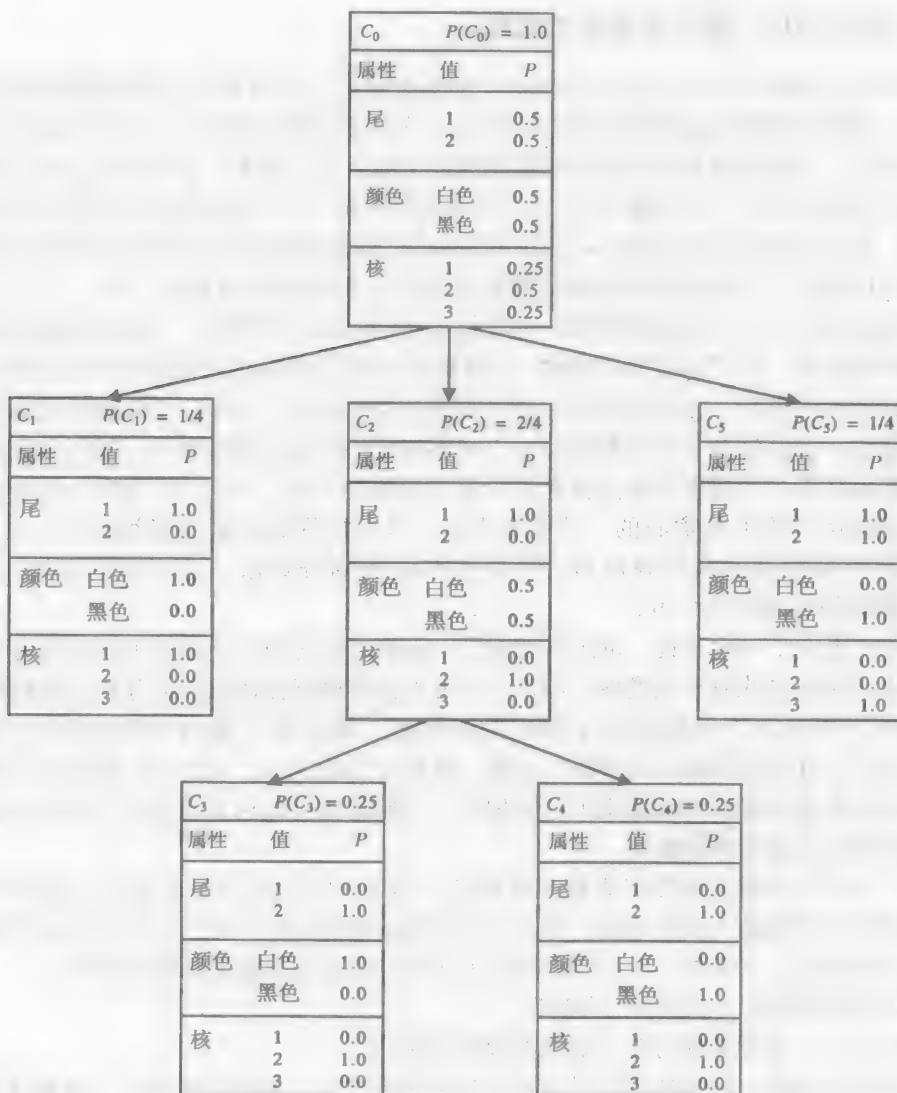


图 11-90 插入第 4 个数据后的节点概率

$$CU(C_1, C_2, \dots, C_k) = (\sum P(C_k) [\sum \sum P(A_i = V_{ij} | C_k)^2 - \sum \sum P(A_i = V_{ij})^2]) / m$$

$$CU = \frac{1}{m} \sum_{k=1}^m P(C_k) \left[\sum_i \sum_j P(A_i = V_{ij} | C_k)^2 - \sum_i \sum_j P(A_i = V_{ij})^2 \right]$$

其中, C_1, C_2, \dots, C_k 表示 k 个簇; 外层的和是对所有的簇求和; 里层的和是所有属性之和; A_i 是第 i 个属性, 它的取值为 V_{i1}, V_{i2}, \dots , 它在所有 j 上求和。

如果你花时间来研究这个公式, 便可以领会这个表达式的意义。生成簇的意义在于, 便于对簇中的实例属性进行预测。也就是说, 与 $P(A_i = V_{ij})$ 相比, 对于簇 C_k 中的属性值 $A_i = V_{ij}$ 的实例, $P(A_i = V_{ij} | C_k)$ 具有更好的概率估计, 这是由于后者考虑了它所在的簇。如果那个信息没有帮助, 则那些簇没有益处。上面的度量计算(多重求和内部)的是在概率平方差方面有帮助信息的数量。这并不是标准方差度量。在内层双重和, 概率平方差在所有属性以及所有可能取值上进行求和。在外层和, 它对所有簇进行加权求和, 权为该簇的概率。

11.11 GCLUTO: 图形化聚类工具箱

GCLUTO(Graphical CLustering Toolkit, 图形化聚类工具)是 GLUTO 数据聚类库的一个图形化前端, 用来对低维和高维数据集进行聚类, 分析不同簇的特性。GCLUTO 提供了三种不同的聚类算法, 直接在对象的特征空间或相似性空间上进行聚类。这些算法分别是基于划分的、凝聚的和图划分的。大多数 GCLUTO 的聚类算法的一个关键特征是把聚类问题看作最优化过程, 它设法最大化或者最小化定义在全局或者局部聚类解空间的特定聚类准则函数。

GCLUTO 提供了七种不同的准则函数来驱动划分和凝聚聚类算法。[6, 7, 8, 9] 对此进行了详述和分析。对于高维数据集, 尤其是正在兴起的文档聚类, 大部分准则函数可以产生高质量的聚类解。除了这些准则函数, GCLUTO 还提供某些传统的局部准则(例如, 单连接、全连接和 UPGMA), 这些准则可以用在凝聚聚类算法中。此外, GCLUTO 还提供基于图划分聚类算法, 非常适合于形成跨越特征空间不同维的邻近区域的聚类。基于划分的、准则驱动的聚类算法的一个重要方面是优化这个准则函数的方法。GCLUTO 采用一种随机增量优化算法, 该算法本质上是贪心的, 计算要求低, 可以得到高质量的聚类解[11]。GCLUTO 的基于图划分聚类算法利用高质量、高效的多级图划分算法, 这些算法来源于 METIS、hMETIS 和超图划分算法[8, 9]。

GCLUTO 提供了一些工具, 可用于理解同一簇的对象之间的关系和不同簇的对象之间的关系, 并以可视化方式发现聚类解。GCLUTO 可以识别最能描述或者区分每个簇的特征。这些特征集可以用来更好地理解每个簇中的对象集, 提供关于簇内容的简洁汇总。此外, GCLUTO 提供了可视化功能来查看簇、对象、特征之间的关系。GCLUTO 算法已经被优化来处理那些对象数量和特征维数都很大的数据集。这些算法可以对具有数以万计的对象、数以千计维数的数据集进行快速聚类。

此外, 由于大多数高维数据集都非常稀疏, 因此 GCLUTO 直接考虑这个稀疏性, 并且存储空间需求大致随输入线性增加。GCLUTO 的目标是使得用户可以以用户友好的图形化方式使用它的聚类能力。此外, GCLUTO 提供了几种方法来交互地显示聚类结果。

GCLUTO 的特征: 它具有如下特征:

- 1) 数据文件、聚类解和可视化显示的投影树视图。
- 2) 详细的对话框, 用来选择聚类选项, 如采用的方法、需要的聚类数、准则函数等。
- 3) 用于查看数据的电子表格界面。





- 4) 用于查看聚类解 HTML 界面。
- 5) 矩阵可视化——彩色的交互矩阵。
- 6) 曲面可视化——采用多维缩放产生的三维可视化显示。


使用 GCLUTO: GCLUTO 简单易用。它本质上是用户友好的, 呈现给用户的信息都简单易懂。用户必须了解各种类型的算法以及不同类型的数据应采用哪种对应算法。如果理解了这些问题, 使用 GCLUTO 便易如反掌。

11.11.1 概述

当聚类数据时, 涉及很多信息项, 例如, 数据文件、聚类解文件和可视化。和许多其他应用一样, GCLUTO 采用工程的概念来组织用户的数据和工作流程。当加载一个工程时, 它的内容将被显示在如图 11-91a) 部分的树形视图中。

工程中的每个项目在树中显示为一个图标。以图标呈现在树结构中的各种项目包括:

- 1) 工程——这表示工程本身。它是工程树的根。一个工程可以包含许多不同的数据项。它用  表示。
- 2) 数据——输入数据到工程之后, 这些图标中的一个将出现在工程树中。对于一个数据, 可能存在许多解, 这取决于用户为聚类数据给出的选项。数据表示为 。
- 3) 解——一个数据项目被聚类后, 将创建一个解项并放置在数据项的下面。解表示为 。
- 4) 矩阵可视化——聚类后产生的一种可视化。所有可视化都出现在生成的解的下面。矩阵可视化表示为 。

5) 曲面可视化 (mountain visualization)——这是另外一种可视化, 用 3D 方式来描述簇之间的相互关系。曲面可视化表示为 。

和窗口树形结构一样, 右击任何项目将出现一个菜单, 显示可以在该项目上执行的操作。双击任何项目将在一个新窗口中打开它的内容, 该窗口被称为视图 (View)。这与图 11-91 中的窗口 b), c) 和 d) 相似。

当在其中一个视图中工作时, 与该视图内容对应的额外菜单选项会出现在菜单栏中。

使用 GCLUTO 聚类包含的步骤

- 1) 首先, 创建一个包含被聚类数据的矩阵或者图文件。
- 2) 从文件打开对话框新建一个工程文件。
- 3) 将数据导入到工程。数据来源于步骤 1 中的矩阵或者图文件。
- 4) 采用给定的各种选项聚类数据。这些选项包括: 采用的方法、需要的簇数目等。通过右击该数据可以打开聚类对话框。一旦完成这些步骤, 你就可以得到解。解在右侧, 给出了簇的详细信息。
- 5) 以矩阵形式和曲面形式通过可视化方式显示解。这可以通过右击解, 并选择矩阵或曲面选项实现。

在接下来的内容中, 我们将详细讲述这些步骤。

创建数据文件: GCLUTO 接受两种格式的数据: 图文件和矩阵文件。它也可以由可选的包含各种对象的列标记和类标记的文件接受数据。

矩阵文件: 这个文件包含矩阵形式的数据。每一行代表一个对象, 列表示对象的维。GCLUTO 以两种格式理解矩阵文件: 稀疏矩阵形式和稠密矩阵形式。矩阵文件以 file_

name.mat 格式存储, 其中 file_name 是文件名。

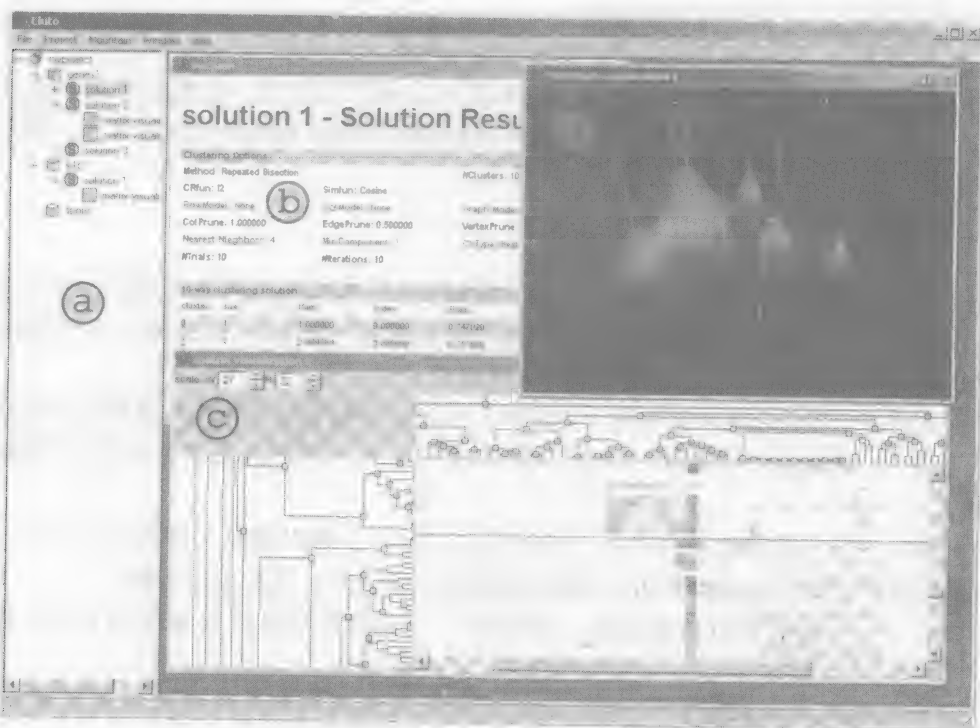


图 11-91 GCLUTO 屏幕截图

稀疏矩阵形式: 稀疏矩阵是矩阵中大多数元素为零的矩阵。具有 m 行 n 列的稀疏矩阵 A 以纯文本文件格式存储, 该文件包含 $m+1$ 行。第一行包含关于矩阵大小的信息, 其余 m 行为真实数据。第一行包含矩阵行数、列数以及非零元素的数目。这些数据必须按照这个顺序给出。每行的非零元素必须被指定为“列-值”对。在 GCLUTO 中的列编号由 1 开始, 而不是像 C、C++、Java 等流行编程语言那样由 0 开始。通常, 列值为整数, 而数据值为实数。表 11-25 给出了一个稀疏矩阵和对应的矩阵文件。

表 11-25 稀疏矩阵

0.5	1.0	0	0	0	0	0	5	7	8	
0	0	2.0	0	0	0.4	0	1	0.5	2	1.0
1.2	0	0	0	0	0	0	3	2.0	6	0.4
0	0	0	-1.0	0	0	0	1	1.2		
-1.0	0	0	0	0	2.3	0	4	-1.0		
							1	-1.0	6	2.3

矩阵文件

矩阵文件

稠密矩阵形式: 稠密矩阵中的多数元素为非零元素。具有 m 行 n 列的稠密矩阵 A 也以纯文本文件存储, 该文件包含 $m+1$ 行。第一行只包含矩阵大小的信息, 即行数和列数。这些数据必须按照这个顺序给出。数据以 $(m \times n)$ 的格式表示, 第 i 个值对应于矩阵 A 的第 i 行。表 11-26 显示了一个稠密矩阵和对应的矩阵文件。

表 11-26 稠密矩阵、矩阵文件

1.0	0.1	0.12	0.54	5	4
1.23	2.1	-0.34	-0.56	1.0	0.1 0.12 0.54
0.34	1.5	0.98	2.1	1.23	2.1 -0.34 -0.56
-0.76	1.2	-0.23	1.2	0.34	1.5 0.98 2.1
0.45	0.43	0.13	1.4	-0.76	1.2 -0.23 1.2
				0.45	0.43 0.13 1.4

矩阵文件

图文件：图文件也包含矩阵形式的数据，但是数据以邻接矩阵形式表示，说明被聚类对象的相似性。矩阵的每一行/列表示一个对象，位置 (i, j) 处的值表示第 i 个对象和第 j 个对象之间的相似性。图文件可以以稀疏图形式和稠密图形式两种格式给出。这两种形式的数据表示与矩阵形式的解释类似，只是这里的矩阵都是方阵。图文件都以`<file_name>.graph`格式存储，其中`file_name`表示存储文件的文件名。

稀疏图形式：拥有 n 个顶点的稀疏图的邻接矩阵 A 被存储在一个包含 $n+1$ 行的无格式文件中。第一行包含矩阵大小信息，其余 n 行包含实际数据。准确地说，第一行包括两个整数，即图中的顶点数和非零元素数。每个顶点的邻接结构被指定为空间上分离的对列表。每个对包含邻接顶点的数量，紧随其后的是对应边的相似性。通常，列值是整数，而数据值是实数值。表 11-27 显示了一个稀疏图和相应的图文件数据。

表 11-27 稀疏图

1.0	0	0	0.2	0	5	8
0	1.0	0	0	0.4	1	1.0 4 0.2
0	0	1.0	0	0	2	1.0 5 0.4
0	0	0	1.0	0	3	1.0
0.6	0	2.3	0	1.0	4	1.0
					1	0.6 3 2.3 5 1.0

图文件数据

稠密图形式：拥有 n 个顶点的稠密图的邻接矩阵 A 被存储在一个包含 $n+1$ 行的无格式文件中。第一行包含矩阵大小信息，其余 n 行包含实际数据。数据以 $(n \times n)$ 的格式表示，第 i 个值对应于矩阵 A 的第 i 列。如表 11-28 所示。

表 11-28 稠密图

1.0	0.6	0.9	0.2	0.9	5
0.2	1.0	0.2	1.6	0.4	1.0 0.6 0.9 0.2 0.9
0.4	0.5	1.0	1.4	0.2	0.2 1.0 0.2 1.6 0.4
0.5	1.2	0	1.0	0.7	0.4 0.5 1.0 1.4 0.2
0.6	0	2.3	0.8	1.0	0.5 1.2 0 1.0 0.7
					0.6 0 2.3 0.8 1.0

图文件

可选文件：为了提供标号作为输入，GCLUTO 提供了三类可选文件，即行标号文件、列标号文件和行类标号文件。

行标号文件：该文件存储每个行的标号。如果矩阵的行数是 m ，那么这个文件应该只包含 m 行。每行所存储的信息被看作一个串，并变成矩阵对应行的标号。也就是说，这个文件的第 i 行包含矩阵的第 i 行的标号。这个文件以格式`<file_name>.mat.rlabel`或者`<file_`

name >. graph. clabel 存储。

列标记文件：该文件存储每列(特征)的标号。如果矩阵中共有 n 列，那么这个文件应该精确地包含 n 行。每行所存储的信息被看作一个串，并变成矩阵对应列的标号。也就是说，这个文件的第 i 行包含矩阵的第 i 列的标号。这个文件以格式 <file_name>. mat. clabel 或者 <file_name>. graph. clabel 存储。

行类标号文件：它包含矩阵行的类标号。如果矩阵中共有 m 行，那么该类标号文件应该只包含 m 行。每行中存储的信息可以被看作一个串，并变为矩阵中对应的对象的类-标号。文件应以 <file_name>. rclass 格式存储。

创建一个新工程：当第一次运行 GCLUTO 时，它以一个空的工程树开始。为了开始工作，必须创建一个新的工程。为了创建工程，定位到菜单栏并选择“File”，然后选择“New Project”。此时将显示一个文件对话框。为工程指定名称和在计算机上的保存位置。

GCLUTO 会创建一个目录，称为工程目录(project directory)。工程目录以工程命名，并被存储在指定位置。在工程目录中，GCLUTO 保存了与工程有关的所有信息。

为打开一个已存在的工程，选择“File”菜单，然后选择“Open Project”。将弹出一个文件对话框。导航到工程目录位置，打开目录内的文件“Project_name. prj”(这里，Project_name 和工程目录名相同)。选择这个文件并单击“Open”。

完成这些步骤后，一个工程便会被加载和显示在工程树中。

导入数据：为导入一个新的数据项，转到菜单栏并选择“Project”，然后选择“Import Data”，将会显示 Import Data 对话框，用户可在该对话框中为上面所列的每种文件类型指定文件位置。

用户可以给数据项指定一个标号。如果没有给出标号，数据项将按照去除扩展名的 *.mat 文件命名。根据数据的文件格式，用户可以提供一个图文件或矩阵文件。除了该文件，用户可以提供标号文件，用来显示解中的标号。点击“Browse”按钮，用户可以提供对应文件。用户还必须选择合适的选项，说明 *.mat 文件包含矩阵数据还是图数据。默认文件选项是 *.mat 文件格式。

如果首先选择 *.mat 文件，GCLUTO 会通过附加扩展文件名到 *.mat 文件名，尽力猜测可选文件(*.rlabel、*.clabel、*.rclass)的位置。

点击 Import Data 对话框的“OK”后，GCLUTO 会试图在选择的文件中读取数据。如果无错误发生，GCLUTO 会在工程树中添加一个新的数据项，并打开一个数据视图。数据视图允许用户观察数据，并验证它是否被正确加载。

聚类数据：如果已用先前步骤将数据导入，则可以进行聚类。聚类可以用两种不同的方式启动。第一种方法是从弹出菜单中选择“Cluster”，右击工程树的数据项将会弹出该菜单。第二种方法是如果数据视图已经打开，可以在“Data”菜单栏下找到相同的菜单。

无论从哪个菜单选择了“Cluster”，将显示 Clustering Option 对话框，该对话框包括所有聚类可用的选项。关于选项的更详尽说明将在后面的内容中给出。为了帮助用户做出明智的选择，在用户进行选择时，GCLUTO 会自动更新对话框，保证只提供合理的选择。

一旦选定了聚类选项，在 Cluster Option 对话框中点击“Cluster”。GCLUTO 完成聚类计算后，会在工程树中创建一个解项，该解项位于聚类数据项的下面。

GCLUTO 还会自动打开一个与图 11-91 的⑥部分类似的解视图。该视图包含聚类所采用的选项和一些关于簇的统计数据。其中一些重要的统计数据是 Isim——簇内元素间的内部相似性，Isdev——簇内元素的标准差，Esim——簇元素与其他元素的外部相似性，Esdev——

簇元素与其他元素的外部标准差。GCLUTO 也提供另外两个重要的统计数据,即描述和区别。一个簇的描述统计详尽描述了簇内特征之间的相似程度。区别统计给出了同一簇内特征之间的差异程度。

可视化:当前, GCLUTO 包含两种可视化: 矩阵可视化 (Matrix Visualization) 和曲面可视化 (Mountain Visualization)。从“solution”菜单可以选择期望得到的可视化, 可视化结果可以由这些解生成。这个菜单可以用如下方法找到: 在工程树中右击解项目, 或者如果此时用户工作在解视图 (Solution View) 下的话, 可以在“Solution”菜单栏下面找到。

矩阵可视化:在矩阵可视化方法中, 原始数据矩阵被显示, 用颜色形象地表示矩阵中的值。GCLUTO 采用白色表示接近零的值, 逐渐加深的红色调表示代表数量大的值, 逐渐加深的绿色表示负值。矩阵的行被重新排序, 使得同一簇的行在一起。黑色水平分割器分离各个簇。如图 11-92 所示。

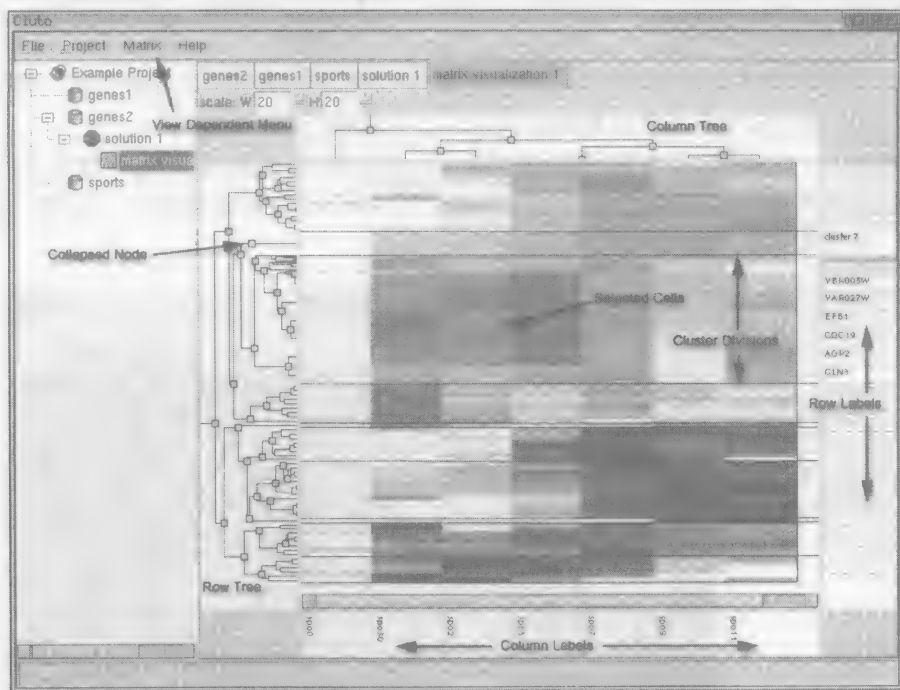


图 11-92 矩阵可视化的屏幕截图

如果选择构建树, 矩阵可视化中将包含树, 它位于矩阵的上方和左部。如果采用了凝聚聚类算法, 在聚类过程中产生的树显示为行树 (Row Tree)。否则, 产生一棵拟合聚类解的树。在矩阵的反转上执行凝聚聚类将产生列树 (Column Tree)。

在导入数据时, 如果选择了行和列标记, 则它们会显示在矩阵的下方和右部。当空间足够时才会显示标号。

为了帮助探查 Matrix Visualization 中包含的信息, 实现了几个特征。首先, 矩阵大小可以用多种方法进行缩放。其次, 树可以用来折叠和展开矩阵中感兴趣区域。

矩阵可视化——缩放:最简单的缩放矩阵的方法是使用位于矩阵上方的比例缩放控件。在文本框中输入一个新的数值, 或者点击向上或向下箭头可以改变缩放比例。标记为“W”的控件控制矩阵的宽度, 标记为“H”的控件影响矩阵高度。这些缩放控件改变矩阵的维, 便

于对矩阵中感兴趣的区域进行缩放。

有时, 用户需要放大矩阵的某一区域, 同时缩小不重要的区域。这类缩放也可以实现。为了调整矩阵某部分的大小, 选择要调整大小的区域。点击任何单元并拖动鼠标到另一个单元完成选择。这两个单元便成为选择区域的两角。被选择单元用蓝色覆盖。为重新调整选择区域的大小, 将鼠标置于所选区域任意边之上, 光标将变为调整大小光标。点击并拖动边到期望位置。被选的所有单元将被重新调整以适应新的区域。

最后, 通过依次选择菜单栏上的“Matrix”→“Reset Sizing”, 矩阵可以恢复到它的原始比例。选择“Matrix”菜单内的“Fit to Screen”可以让矩阵自动缩放以适合屏幕的大小。

矩阵可视化——使用树: 行树和列树可用于折叠(collapsing)和展开(expanded)矩阵。树中蓝色方块表示完全展开的节点。点击任意一个展开的节点可以折叠该节点。折叠的节点以粉红色方块表示。当折叠一个节点的时候, 其所有后代均被隐藏。如果行树中的一个节点被折叠, 那么折叠区域的所有行均被隐藏, 并以包含这些行的均值的一行将其替代。仅需点击一个折叠节点即可将其再次展开。列树的工作方式与此类似。

标记将变化以描述被折叠的区域。如果区域包含的行都属于一个相同的簇, 则它将用该簇的 ID 标记。如果多个簇出现在被折叠的区域, 它将被标记为“多簇”。

曲面可视化: 曲面可视化(见图 11-93)用于可视化聚类间的相对相似性以及它们的大小、内部相似性和内部偏差。在曲面可视化中, 每个簇表示为 3D 地形中的一个山峰。山峰的位置、容量、高度和颜色均用于描绘其相关的簇的信息。



图 11-93 曲面可视化截图

用户能够通过 3D 显示中点击和拖动鼠标进行 3D 可视化的导航。不同的鼠标键完成不同的动作:

- 点击左键: 旋转

- 点击右键：上移、下移、左移和右移
- 点击中键：缩放

这些山峰在平面中的位置通过在每个簇中点上的多维缩放(MDS)来决定。MDS 试图在顶点从高维空间映射到低维空间时保留顶点之间的距离信息。在本应用中，簇中点作为 MDS 中的顶点并映射到一个二维平面。

MDS 允许用户使用曲面可视化对其数据进行推理。例如，在图 11-93 中，一个数据矩阵聚类为 10 个簇。曲面可视化将 10 个簇表示为 10 个山峰并以其簇 ID 标记。尽管要求产生 10 个簇，但是 MDS 将这些山峰放置在 2 个不同的组群中。我们可以推测出每个组群中的簇相似性极高，而分置于不同组群的簇之间具有很大的差异性。因而，可视化的结果显示数据最好地被二路聚类(two-way clustering)。

每个山峰的形状都是高斯曲线。这种形状用于对每个簇中的数据分布进行粗略估计。每个山峰的高度和簇内部的相似性成正比。山峰的体积和每个簇中的元素个数成正比。产生的高斯曲线进行叠加形成曲面可视化区域。

注意：在比较山峰的高度时请牢记曲面可视化已经对粉红色曲线进行了叠加。如图 11-94 所示，最终生成的高度比实际高度要高。

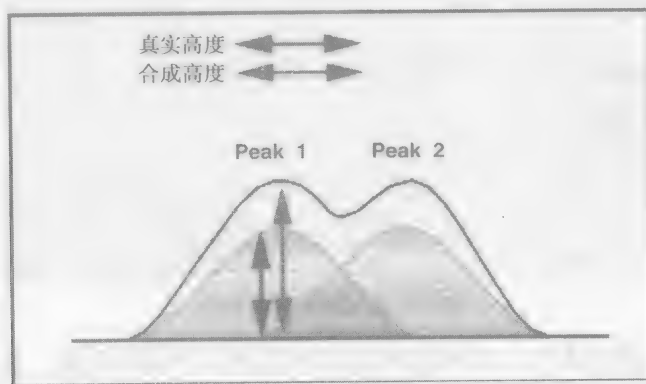


图 11-94 如何叠加顶峰的高度

山峰的颜色和簇的内部偏差成正比。红色表示低偏差，蓝色表示高偏差。只有山峰的顶部的颜色是重要的。在其他部分，其颜色决定于混合颜色的平滑过渡。

点击任意标记将在可视化窗口之下的文本窗口中载入相关簇的统计信息。这个信息和解报告(Solution Report)中的信息相同。如果选择列标记作为数据的标识，则曲面可视化能够在每个山峰的上部显示最常见的特征。这一选项称为“Show Features”，可在“Mountain”菜单中找到该选项。

11.11.2 GCLUTO 中的可用选项

在导入的数据上单击鼠标右键并选择聚类，将得到如图 11-95 所示的界面。根据所选择方法的不同，可以获得其他针对这一方法可用的选项。

聚类方法。GCLUTO 提供了四种聚类方法。

反复二分法。在这种方法中，期望的 k 路聚类通过 $k-1$ 次反复二分实现。使用这种方法，矩阵首先聚类为两组，然后选中其中一组并进一步二分。继续这一过程，直到得到期望的簇个数为止。二分聚类以如下方式进行，即在每一步，优化一个特定聚类函数(由用户给

定)。被选择进行进一步划分的簇在聚类选择下拉菜单中给出。

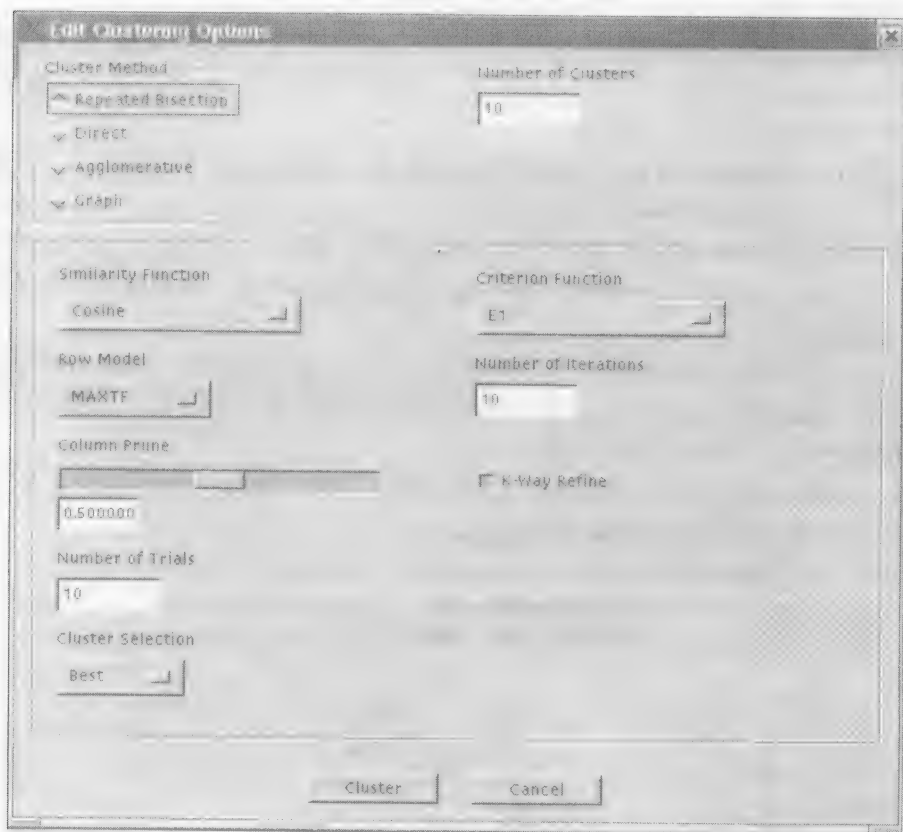


图 11-95 聚类选项界面

直接方法。在这种方法中， k 路聚类通过同时找到 k 个簇来实现。这种方法比反复二分法慢。对于 k 值小于 10~20 的情况，使用这种方法可以获得更好的结果。对更高的 k 值，反复二分法是更好的方法。

凝聚方法。这种方法是自底向上的方法。数据点最初被当作是单独的簇，然后通过使用某种准则函数，这些单独的簇被一步步聚合，直到得到期望的簇为止。凝聚法有四种聚类函数：单链接(single link)、完全链接(complete link)、加权简单链接(weighted simple link)和 UPGMA。

图方法。在这种方法中，期望的 k 路聚类通过首先获得一个最近邻图，其中每个对象被当作一个顶点，而每个对象和其最为相似的其他对象相连接来实现。然后，通过使用最小割(minicut)图划分算法将图划分为 k 个簇。

所示的屏幕中显示了使用反复二分法的各种选项：簇个数、相似性函数、准则函数、行模型、迭代次数、列裁剪等。下面我们讨论每种方法的各种不同选项。

所有方法的公共选项

相似函数。GCLUTO 提供了三种相似度函数，即余弦、相关系数和欧式距离。

余弦选项。对象间的相似度通过计算向量的余弦函数得到。这是默认值。

相关系数。对象间的相似度通过计算向量的相关系数得到。

欧式距离。对象间的相似度是和它们之间的欧式距离负相关的。

两个对象 i 和 j 之间的相似度通过下式给定:

$$\text{sim}(i, j) = 1 - \frac{d_{ij}}{1.0 + d_{\max}}$$

其中, d_{\max} 是数据集中任意两个对象间的最大距离, d_{ij} 是第 i 个对象和第 j 个对象间的距离。

簇个数。这是一个整数, 指定数据将被划分成的簇的个数。

行模型。行模型选择用于缩放每行各列的模型。可能值为 none、maxtf、sqrt 以及 log。

none。列值即为输入文件所提供的值。这是默认值。

maxtf。每行的列值缩放到 0.5 和 1.0 之间。

sqrt。每行的列值为其实际值的平方根。

log。每行的列值为其实际值的对数值。

列裁剪。一般而言, 并非所有的维都对数据的聚类有用。根据 Pareto 法则, 只有一个很小的维集合对聚类有贡献, 也就是说, 只有很少的维是真正起作用的。因此, 可以在不影响最终结果的前提下删除一些维。这个值指定了在进行聚类之前需要删除的列所占的比例。这个值应该在 0.5 和 1 之间。总体而言, 这个参数可以在不严重影响总体聚类质量的前提下, 大大减少列的个数。

在直接方法中可用的附加选项。直接方法在五个基本选项的基础上提供了三个附加选项, 它们是: 迭代次数、试验次数和准则函数。

迭代次数。这个值表示在聚类的每一步中进行的精炼迭代次数。合理的取值在 5 ~ 20 之间。默认值为 10。

试验次数。这个值指定了要计算的聚类解的个数。如果给定的值为 n , 那么 GCLUTO 将使用不同的初始化种子, 计算 n 个不同的聚类解, 然后在其中选择最好的解。默认值为 10。

准则函数。基本上, GCLUTO 提供了七种准则函数来产生聚类, 它们是 i_1 , i_2 , e_1 , g_1 , g_1' , h_1 , h_2 。数学公式见表 11-29。

表 11-29 GCLUTO 的准则函数和优化函数列表

准则函数	优化函数
I_1	$\text{maximize} \sum_{i=1}^k \frac{1}{n_i} \left(\sum_{v, u \in S_i} \text{sim}(v, u) \right) \quad (1)$
I_2	$\text{maximize} \sum_{i=1}^k \sqrt{\sum_{v, u \in S_i} \text{sim}(v, u)} \quad (2)$
E_1	$\text{maximize} \sum_{i=1}^k n_i \frac{\sum_{v \in S_i, u \in S_i} \text{sim}(v, u)}{\sqrt{\sum_{v, u \in S_i} \text{sim}(v, u)}} \quad (3)$
G_1	$\text{maximize} \sum_{i=1}^k \frac{\sum_{v \in S_i, u \in S_i} \text{sim}(v, u)}{\sqrt{\sum_{v, u \in S_i} \text{sim}(v, u)}} \quad (4)$
G_1'	$\text{maximize} \sum_{i=1}^k n_i^2 \frac{\sum_{v \in S_i, u \in S_i} \text{sim}(v, u)}{\sqrt{\sum_{v, u \in S_i} \text{sim}(v, u)}} \quad (5)$
H_1	$\text{maximize} \frac{I_1}{E_1} \quad (6)$
H_2	$\text{maximize} \frac{I_2}{E_1} \quad (7)$

- 1) i_1 ——选择准则函数 I_1 。该函数试图最大化簇中元素之间的簇内相似度。
- 2) i_2 ——选择准则函数 I_2 。这个函数也试图最大化簇中元素之间的簇内相似度。唯一的不同之处在于这里取的是函数的平方根。
- 3) e_1 ——选择准则函数 E_1 。这个函数将簇间相似度除以簇内相似度的平方根。
- 4) g_1 ——选择准则函数 G_1 。这个函数除了分母不取平方根之外其他部分和 E_1 相同。
- 5) g_1' ——选择准则函数 G_1' 。这个函数除了增加 n_i^2 和分母不取平方根之外其他部分也和 E_1 相同。
- 6) h_1 ——选择准则函数 H_1 。这是一个混合函数, 试图最大化 I_1/E_1 。
- 7) h_2 ——选择准则函数 H_2 。这是一个混合函数, 试图最大化 I_2/E_1 。

反复二分法可用的附加选项。反复二分法在四个基本选项的基础上提供了四个附加选项, 它们是迭代次数、准则函数、试验次数和簇选择。迭代次数、准则函数以及试验次数和直接方法相同。

簇选择。GCLUTO 在重复划分的方法中为用户提供两种方法选择簇。这两种方法是 Best (最好) 和 Large (最大)。

最好。选择可以使总体聚类函数值最优的簇进行二分。这是默认值。

最大。选择当前最大的簇在下一步中进行二分。

凝聚法中可用的附加选项。凝聚法在四个基本选项的基础上提供了一个附加选项。这个选项是准则函数。

准则函数。凝聚法使用四个函数将个体簇分组, 直到得到满足要求的簇个数。它们是:

Wslink: 选择加权单链接准则函数。

Clink: 选择传统完全链接准则函数。

Wclink: 选择聚类加权完全链接准则函数。

Upgma: 选择传统 Upgma 准则函数。

图方法中可用的附加选项。图方法在四个基本选项的基础上提供了七个附加选项。它们是:

1) 最近邻。这个参数指定每个对象的最近邻个数并用于生成最近邻图, 进而用于基于图划分的聚类算法。

2) 边裁剪。该参数用于消除最近邻图中的某些可能将不同簇中顶点相连的边。

3) 顶点裁剪。该参数用于消除最近邻图中某些可能是离群点的顶点。

4) 最小连通分支。该参数用于在聚类前从最近邻图中消除小连通分支。

5) 图模型。该参数控制实时构建并提供给基于图划分聚类算法的最近邻图的类型。有四种图模型, 即直接对称、直接非对称、对称链接和非对称链接。

6) 直接对称。图以如下方式构建, 即两个对象 u 和 v 之间存在一条边, 当且仅当二者存在于彼此的最近邻列表中。也就是说, u 是 v 的最近邻, 反之亦然。

7) 直接非对称。图以如下方式构建, 即两个对象 u 和 v 之间存在一条边, 只要一个对象存在于另一个对象的最近邻列表中。也就是说, v 是 u 的最近邻之一或 u 是 v 的最近邻之一。

8) 对称链接。图以和直接对称同样的邻接结构构建。但是, 每条边 (u, v) 的权重等于 u 和 v 的近邻节点列表中的共同顶点个数。这一选项由 CURE 算法所使用的链接图得来。

9) 非对称链接。图以和直接非对称相同的邻接结构构建。

问题: 聚类美国的大城市

表 11-30 为美国 49 个大城市的人口统计数据[12]。例如, 亚特兰大市人口的 67% 是非洲裔美国人, 2% 是西班牙裔, 1% 为亚洲裔。平均年龄为 31 岁, 失业率为 5%, 人均收入为 \$22000。现在, 按人口统计数据的相似性将 49 个城市聚类为 4 个簇。

步骤 1: 创建矩阵文件。在 MS Excel 中输入该文本, 不带列名(人口统计数据)和行名(城市)。将该文件命名为 america。在存放文件时, 从“save as”对话框中选择“Formatted Text (Space delimited)”。系统将显示某种警告, 只需要确认。

现在, 文件类型将为 America. prn。将该文件重新命名为 america. mat。可以打开(包含该文件的)文件夹, 右击该文件, 选择重新命名选项来完成重命名工作。在控制台状态, 使用 ren datal. prn datal. mat 命令(Windows 下)或 mv datal. prn datal. mat 命令(Linux 下)。

然后, 在一个独立的文本文件中逐行输入城市名, 并将该文件命名为 america. mat. rlabel。

接着, 创建另一个名为 america. mat. clabel 的文本文件, 逐行输入列字段。

后续, 生成另一个文件包含一行行的列。

表 11-30 美国 49 个大城市的人口统计数据

城市	非洲裔	西班牙裔	亚洲裔	平均年龄值	失业率	人均收入
阿尔伯克基	3	35	2	32	5	18
亚特兰大	67	2	1	31	5	22
奥斯汀	12	23	3	29	3	19
巴尔的摩	59	1	1	33	11	22
波士顿	26	11	5	30	5	24
夏洛特	32	1	2	32	3	20
芝加哥	39	20	4	31	9	24
辛辛那提	38	1	1	31	8	21
克里夫兰	47	5	1	32	13	22
哥伦布	23	1	2	29	3	13
达拉斯	30	21	2	30	9	22
丹佛	13	23	2	34	7	23
底特律	76	3	1	31	9	21
艾尔帕索城	3	69	1	29	11	13
沃思堡	22	20	2	30	9	20
弗雷斯诺	9	30	13	28	13	16
火奴鲁鲁	1	5	71	37	5	24
休斯顿	28	28	4	30	7	22
印第安纳波利斯	22	1	1	32	5	21
杰克逊维尔	25	3	2	32	7	19
堪萨斯城	30	4	1	33	6	21
拉斯维加斯	11	13	4	33	5	20
长滩	14	24	14	30	8	21
洛杉矶	14	40	10	31	11	21
孟菲斯	55	1	1	32	9	20
迈阿密	27	63	1	36	12	17
密尔沃基	31	6	2	30	5	22
明尼阿波利斯	13	2	4	32	5	23
纳什维尔	23	1	1	33	3	24
新奥尔良	62	4	2	32	7	18
纽约	29	24	7	34	11	27
奥克兰	44	14	15	33	10	24

(续)

城市	非洲裔	西班牙裔	亚洲裔	平均年龄值	失业率	人均收入
俄克拉何马城	16	5	2	32	6	17
奥马哈	13	3	1	32	5	20
费城	40	6	3	33	9	23
凤凰城	5	20	2	31	4	19
匹兹堡	26	1	2	35	7	21
波特兰	8	3	5	35	7	20
萨克拉门托	15	16	15	32	8	20
圣路易斯	48	1	1	33	8	23
圣安东尼奥	7	56	1	30	5	17
圣地亚哥	9	21	12	31	8	20
三藩市	11	14	29	36	6	31
圣何塞	5	27	20	30	8	26
西雅图	10	4	12	35	5	28
托莱多	20	4	1	32	6	19
图森	4	29	2	31	3	19
塔尔萨	14	3	1	33	4	20
弗吉尼亚海滩	14	3	4	29	6	18

步骤 2: 创建一个工程。在命令行上输入 `geluto` (如果是 Linux) 或运行 `geluto.exe` (如果在 Window 环境下)。

这时会出现一个空窗口。通过点击文件以及新工程选项创建一个工程。将工程命名为 `America`。现在可以看到 `America` 以工程图标显示。

步骤 3: 导入数据。现在右击 `America` 并选择导入数据。系统会弹出一个对话框。给定标号名为 `America-data`。然后, 选择矩阵文件并使用浏览器定位到步骤 1 中创建的 `america.mat` 文件。由于已经创建了 `america.mat.rlabel` 和 `america.mat.clabel` 文件, 因此这些特征值也会关联显示。但是, 这些文件必须和数据文件在同一个路径中。如果正确执行, 可以看到文件所列的数据值。

步骤 4: 聚类数据。右击 `America-data` 并选择 `Cluster` 选项, 可以看到一个聚类对话框。选择聚类方法 (cluster method) 为凝聚法 (Agglomerative)。给定聚类个数 (number of cluster) 为 4, 选定相似性函数 (similarity function) 为余弦 (cosine)。选定条件函数为 `UPGMA`, 选定行模型 (row model) 为空, 列裁剪 (column prune) 为 1.0, 保留试验次数 (number of trials) 和迭代次数 (number of iteration) 为 10。这样可以给出答案。

步骤 5: 生成矩阵视图。现在选择解答 (solution) 并右击选择矩阵视图。给定特征数为 18, 选择构建行树选项。点击确定 (ok) 即可得到矩阵视图。

在图 11-96 中, 通过点击左边的小方框可以查看属于各个簇的城市。继续点击直到仅有四个方框为止 (从内部的方框向外部的方框点击, 以便看到实际的簇合并过程)。

步骤 6: 产生曲面视图。现在选择解答并右击选择曲面可视化来得到曲面视图。

从图 11-97 可以看到簇 0 只有一个元素, 并且深蓝色表示特征间的内部偏差很高。簇 2 顶部的深红色表示特征间的内部偏差很低, 并且簇 2 的宽广的扩展度表示大量元素 (城市) 落于其类别中。对其他簇而言, 也可以得到相似的结论。



图 11-96 矩阵可视化输出

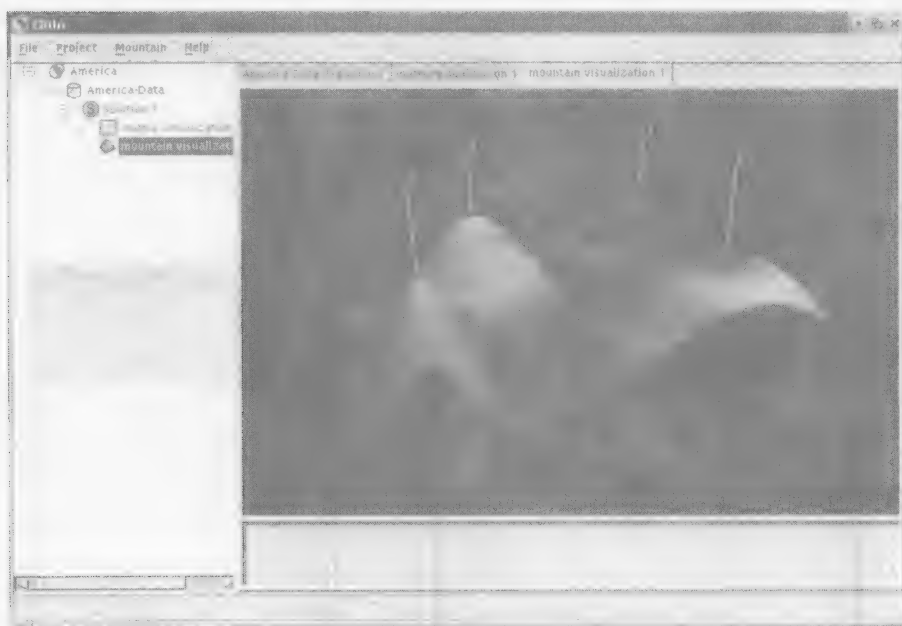


图 11-97 曲面可视化输出

11.11.3 使用 GCLUTO 进行文本挖掘

我们考虑使用文本文件 `textdata.txt` 中的数据(见图 11-98)。

现在,使用 `doc2mat` 工具将这一文本文件转化为矩阵文件。从网站下载这个工具并在命令行转到该目录,并在命令提示符下输入 `> doc2mat <textfile name> <matrix filename>`。

例如,可以在命令行输入 `> doc2mat textdata.txt textdata.mat` 将文本数据文件转化为矩阵

文件。

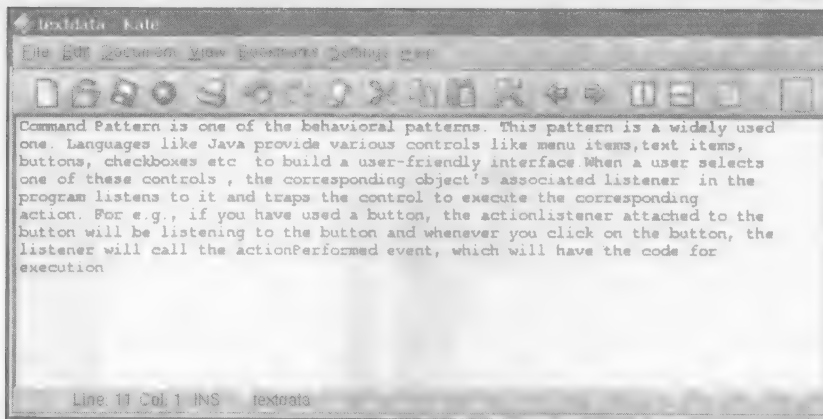


图 11-98 文本文件 textdata.txt

textdata.mat 文件的第一行给出了行数、列数和非负值的个数。矩阵文件具有稀疏矩阵的格式。textdata.mat.clabel 文件包含矩阵的列标识,即文件中出现的唯一单词,基于这些单词进行聚类。textdata.mat.clabel 文件表示的特征由数字 1, 2, ... 表示。wide 由 1 表示, pattern 由 2 表示, 等等。下面说明如何产生 .mat 文件。从文件中取第一行, 即

“Command pattern is one of the behavioural patterns. This pattern is widely used.”

现在, doc2mat 工具从中识别出 4 个关键词, 即 wide、pattern、behaviour 和 command。pattern 出现了三次, 而其他关键字各出现了一次。

因此, 稀疏矩阵的第一行可以表示为:

1 1 2 3 3 1 4 1

类似地, 可以对其他行进行表示(见图 11-99)。

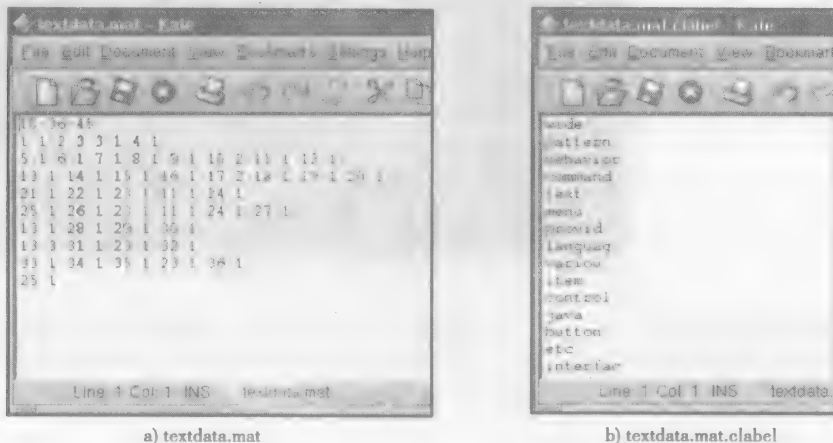


图 11-99 文本文件

一旦生成 mat 文件之后, 打开 CCLUTO 并按照以前的步骤生成聚类。这里需要做一个改变, 即对一个新的选项(也就是列模型)进行设置, 其值为词频的倒数。给定簇的个数后点击聚类对话框中的聚类按钮, 就会显示结果(对这个例子而言, 簇的个数为 4)。

最后的结果应该如图 11-100 所示。

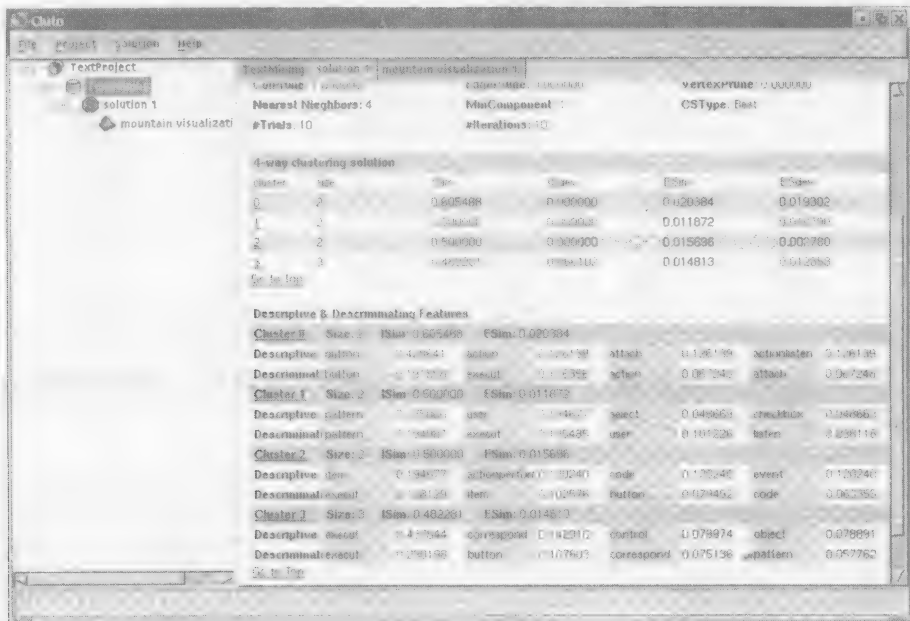


图 11-100 输出格式

从输出可以推断,对簇 0 而言,button 特征和该簇中的其他特征具有 42.86% 的平均相似性,并且和该簇中其他特征具有 19.15% 的相异度。还可以得到聚类的曲面视图,并对聚类进行分析。如果对“itme”单词进行查询,该词被分类到簇 2 的概率为 19.46%。类似地,对 button 单词查询,其被分类到簇 0 的概率为 42.86%。

即使 GCLUTO 提供了关于文本聚类的一些推断,其对最终结果并没有提供足够的信息。也就是说,用户必须对这些单词的结果进行解释。而且,对聚类类似于新闻组(newsgroup)的数据集,使用含有 Rainbow 程序的工具包 Bow 是非常合适的。

习题

- 给定 6 维分类样本 $C=(A, B, A, B, A, A)$, $D=(B, B, A, B, B, A)$, 试求:
 - 1) 样本间相似性的简单匹配系数(SMC)。
 - 2) Jaccard 系数。
 - 3) Rao 系数。
- 给定一个 5 维分类样本集合

$A=10110$

$B=11010$

$C=00110$

$D=01010$

$E=10101$

$F=01101$

1) 应用凝聚层次聚类并使用

① 基于 Rao 系数的单链接相似度量。

②基于简单匹配系数 SMC 的完全链接相似度量。

2) 给出上题中问题①和②答案的树状图。

3. 画出树状图来显示 7 种蛋白质之间的相似度(距离相似度)。使用完全链接方法。

	1EQR	1ATI	1ADJ	1EFW	1ASZ	1B8A	1BBW
1EQR	0	0.21	0.23	0.55	0.28	0.31	0.31
1ATI	0.21	0	0.25	0.24	0.24	0.18	0.21
1ADJ	0.23	0.25	0	0.24	0.24	0.21	0.23
1EFW	0.55	0.24	0.24	0	0.34	0.36	0.3
1ASZ	0.28	0.24	0.24	0.34	0	0.41	0.27
1B8A	0.31	0.18	0.21	0.36	0.41	0	0.28
1BBW	0.31	0.21	0.23	0.3	0.27	0.27	0

4. 使用下表给出的相似性矩阵进行单链接以及完全链接层次聚类。使用树状图显示结果。在树状图中展示节点合并的顺序。

相似性矩阵					
	P_1	P_2	P_3	P_4	P_5
P_1	1	0.1	0.41	0.55	0.35
P_2	0.1	1	0.64	0.47	0.98
P_3	0.41	0.64	1	0.44	0.85
P_4	0.55	0.47	0.44	1	0.76
P_5	0.35	0.98	0.85	0.76	1

5. 五个人在食物(X_1)和衣服(X_2)上的每日支出额见下表:

(数值是假想的, 并非实际数值)。使用单链接、完全链接以及平均链接方法对上述数据聚类并比较其树状图。使用欧式距离作为距离度量。

6. 下表给出了两个变量上的六个观测样本:

个人	X_1	X_2
A	2	4
B	8	2
C	9	3
D	1	5
E	8.5	1

对象	X_1	X_2
a	3	2
b	4	1
c	2	5
d	5	2
e	1	6
f	4	2

1) 以散点图绘制观测样本。判断有多少分组, 以及每个分组有多个成员。

2) 应用最近邻方法并以欧式距离作为相异性度量。使用树状图得到分组个数及其成员。

3) 和问题 2 相同, 但使用最远邻方法(完全链接方法)。

4) 和问题 2 相同, 但使用平均链接方法。

5) 使用 k-均值方法, 假定观测点分为两组, 其中一组包括 a 和 e。

7. 下表给出了两个变量的六个观测样本:

对象	X_1	X_2	对象	X_1	X_2
a	-1	-1	d	-2	-2
b	0	0	e	1	-1
c	2	2	f	1	2

- 1) 以散点图绘制观测样本。判断有多少分组, 以及每个分组有多少个成员。
 - 2) 应用最近邻方法并以欧式距离作为相异性度量。使用树状图得到分组个数及其成员。
 - 3) 和问题 2 相同, 但使用最远邻方法(完全链接方法)。
 - 4) 和问题 2 相同, 但使用平均链接方法。
 - 5) 使用 k 均值方法, 假定观测点分为两组, 其中一组包括 a 和 e 。
8. 一个高保真音响爱好者杂志对 19 个品牌的中型扩音器进行了测试。这些扩音器的测试结果及其价格列于下表:

品牌	价格	准确度	贝斯	功率
A	600	91	5	38
B	598	92	4	18
C	550	90	4	36
D	500	90	4	29
E	630	90	4	15
F	580	87	5	5
G	460	87	5	15
H	600	88	4	29
I	590	88	3	15
J	599	89	3	23
K	598	85	2	23
L	618	84	2	12
M	600	88	3	46
N	600	82	3	29
O	600	85	2	36
P	500	83	2	45
Q	539	80	1	23
R	569	86	1	21
S	680	79	2	36

“价格”是制造商的建议零售价(单位为美元)。“准确度”从 0 ~ 100, 是扩音器再现乐谱中的每个频率的能力。“贝斯”从 1 ~ 5, 是扩音器处理大贝斯音符的程度。“功率”是扩音器需要的再现中等音量音乐的每频道最小扩音功率(单位为瓦特)。该杂志需要将这些品牌进行同类和异类分组。你对该杂志有何建议?

9. 下表汇总了从 1982 年到 1993 年间 Bradford 二月份的气象数据。变量本身非常直观。这些数据是否存在明显的聚类, 即哪些年份的二月份天气比较相似?

年份	平均温度	最高温度	最低温度	土壤温度 (10cm 处)	月降雨量 (毫米)	日最大 降雨量	降雪天数
1982	4.2	13.3	-5.3	4	23	6	0
1983	1	7.8	-5.3	3	34	11	8
1984	2.9	11.4	-5.1	3.2	65	17	0
1985	1.6	10.2	-6	2.9	7	2	5
1986	-1.1	2.7	-9	1.5	22	5	24
1987	3.3	13.4	-7.3	2.7	46	15	2
1988	4.5	13	-2.9	3.7	89	22	4
1989	5.7	13.5	-2.7	5.2	92	16	0
1990	6.6	14.9	-0.6	5.5	131	29	0
1991	1.5	13.8	-7.2	2.6	80	18	14
1992	5.5	12.1	-3.6	4.5	46	9	2
1993	4.7	12	-3	5.1	14	6	0

10. 下面的数据是 Rataj 和 Schindler(1991, Binary, 159 – 164)描述的数据集的一部分。数据中包含六个物种, 大多数(物种)有超过 1 个种类和 16 个表型特征的数据(0 = 缺失, 1 = 存在)。这些物种为:

- *ecoli*(大肠杆菌)
- *styp*hi(沙门氏菌伤寒菌)
- *kpneu* *Klebsiella pneumoniae*
- *pvul*(变形杆菌)
- *pmor* *P. morganii*
- *smar* *Serratia marcescens*

Species	H2S	MAN	LYS	IND	ORN	CIT	URE	ONP	VPT	INO	LIP	PHE	MAL	ADO	ARA	RHA
<i>ecoli</i> 1	0	1	1	1	0	0	0	1	0	0	0	0	0	0	1	1
<i>ecoli</i> 2	0	1	0	1	1	0	0	1	0	0	0	0	0	0	1	0
<i>ecoli</i> 3	1	1	0	1	1	0	0	1	0	0	0	0	0	0	1	1
<i>styp</i> hi1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0
<i>styp</i> hi2	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
<i>styp</i> hi3	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0
<i>kpneu</i> 1	0	1	1	1	0	1	1	1	1	1	0	0	0	1	1	1
<i>kpneu</i> 2	0	1	1	1	0	1	1	1	1	1	0	0	1	0	1	1
<i>kpneu</i> 3	0	1	1	1	0	1	1	1	1	1	0	0	1	1	1	1
<i>kpneu</i> 4	0	1	1	1	0	1	1	1	0	1	0	0	1	1	1	1
<i>kpneu</i> 5	0	1	1	1	0	1	0	1	1	1	0	0	1	1	1	1
<i>pvul</i> 1	1	0	0	1	0	1	1	0	0	0	0	1	0	0	0	0
<i>pvul</i> 2	1	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0
<i>pvul</i> 3	1	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0
<i>pmor</i> 1	0	0	1	1	1	0	1	0	0	0	0	1	0	0	0	0
<i>pmor</i> 2	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0
<i>smar</i>	0	1	1	0	1	1	0	1	1	0	1	0	0	0	0	0

使用二元相似性度量以及单链接方法对数据进行聚类。并画出树状图。

11. 下面是一个简单的数据集, 要求给出三种不同的距离矩阵。然后使用单链接聚类方法构建树状图。

实例	x	y
1	1	2
2	3	2
3	6	6
4	10	7
5	8	8

12. 一个消费者机构每年对其会员进行调查。去年的调查问题包括请其会员评价 42 个全国连锁酒店的卫生状况、床位舒适度等情况。*hotels.dat* 文件(在随书光盘中 CHAPTER11 文件夹中给出)包含了数千个回复的汇总, 部分数据见下表:

酒店编号	价格	卫生状况	房间大小	床位舒适度	气氛控制	噪声	设施	服务
1	36	3	3	3	3	3	3	3
2	36	1	2	1	1	1	1	1
3	37	2	2	2	1	1	2	3

(续)

酒店编号	价格	卫生状况	房间大小	床位舒适度	气氛控制	噪声	设施	服务
.....								
.....								
42	129	4	3	4	4	4	4	4

“价格”是会员支付的平均价格，四舍五入按美元计。其他各列是会员对每个特征的平均评价，取值从1(差)~5(最好)，结果四舍五入成整数。将这42个酒店按质量进行分组(如差、可接受、好、很好、最好)。质量和价格之间是否有关系?

13. 使用不同的链接规则对下表数据进行聚类，画出树状图。

	水	蛋白质	脂肪	乳糖	灰分
美洲野牛	86.9	4.8	1.7	5.7	0.9
水牛	82.1	5.9	7.9	4.7	0.78
骆驼	87.7	3.5	3.4	4.8	0.71
猫	81.6	10.1	6.3	4.4	0.75
鹿	65.9	10.4	19.7	2.6	1.4
狗	76.3	9.3	9.5	3	1.2
海豚	44.9	10.6	34.9	0.9	0.53
驴	90.3	1.7	1.4	6.2	0.4
大象	70.1	3.6	17.6	5.6	0.63
狐狸	81.6	6.6	5.9	4.9	0.93
豚鼠	81.9	7.4	7.2	2.7	0.85
河马	90.4	0.6	4.5	4.4	0.1
马	90.1	2.6	1	6.9	0.35
美洲骆驼	86.5	3.9	3.2	5.6	0.8
猴子	88.4	2.2	2.7	6.4	0.18
骡子	90	2	1.8	5.5	0.47
猩猩	88.5	1.4	3.5	6	0.24
猪	82.8	7.1	5.1	3.7	1.1
兔子	71.3	12.3	13.1	1.9	2.3
老鼠	72.5	9.2	12.6	3.3	1.4
驯鹿	64.8	10.7	20.3	2.4	1.4
海豹	46.4	9.7	42	0	0.85
绵羊	82	5.6	6.4	4.7	0.91
鲸鱼	64.8	11.1	21.2	1.6	1.7
斑马	86.2	3	4.8	5.3	0.7

14. 使用 EXCEL 生成一个数据文件：行数 = 400，特征数 = 20，数据在 10 ~ 20 之内。生成一个包含特征值(如 A, B, C, D, ...)的列标识文件。使用直接聚类方法将数据分为 5 个簇。
15. 生成一个稀疏图文件并使用图方法将数据分为 10 个簇。对各种组合进行尝试并分析结果。
16. 生成一个文本文件，它包括以下文本：

“Good morning! I am trying to find out what is the use of GCLUTO and how it is done. GCLUTO is one of the graphical clustering software used to cluster a given data. It has a nice graphical output wherein one can visualize the output in a graphical format. Just check this out. GCLUTO, GCLUTO, GCLUTO is one of the best clustering toolkits available in this world. Play and learn with GCLUTO.”

使用 doc2mat 工具将该文本文件转化为矩阵文件。使用凝聚方法将数据分为 10 个簇并分

析下列词会被聚类到哪一个簇(给出相应的概率值)。

- 1) morn
- 2) cluster
- 3) output
- 4) GCLUTO

WEKA 练习

概念聚类

使用 weather.arff 数据集并选择 Cobweb 聚类算法。有两个重要选项: Cutoff(默认值 = 0.002)和 Acuity(默认值 = 1.0)。首先使用默认值。像使用简单 k 均值一样,对聚类模式使用相同的设置初始化 Cobweb 算法。输出面板给出了合并和分裂的节点个数、簇的个数以及形成的分层聚类结构。对分层聚类结构进行评价。

- 尝试提供 Cutoff 参数值,以抑制节点的分隔和簇的个数。
- 对这个数据集,最好的 Cutoff 值是什么?以树结构画出分层聚类结构。
- 使用可视化工具对簇和簇的实例(节点)进行分析。
- 是否可以检测到明显的异常,如相似的训练样本被聚类到树的完全不同的部分。

对于数值属性,其分组计算是基于均值和方差的(见 Witten & Frank, 2000, 217 ~ 218 页)。当单个实例被指派到一个簇时,属性值上附加了一个最小方差以避免零方差。这个最小方差是由 Acuity 参数设定的。讨论其对具有数值属性的数据集的影响。

COBWEB 练习

- 从随书光盘中打开 iris.arff 文件并保存到你的计算机上。
- 对每个类(山鸢尾类、变色鸢尾类、弗吉尼亚鸢尾类)保留 5 个训练样本,删除其他的数据,产生一个有 15 个数据的训练集。命名为 iris-cobweb1.arff 并保存。
- 在该数据集上应用 COBWEB(使用默认值, Cutoff = 0.002, Acuity = 1)。你会再次发现形成了太多的簇。
- 尝试减小 Acuity 的值以在该数据集上得到更低的误差率。
- 使用可视化工具对聚类及其实例进行分析。

Cobweb 的一个主要问题是对训练样本数据的顺序敏感。使用 iris-cobweb1.arff 生成一个 iris-cobweb2.arff 文件,改变文件中鸢尾植物的三个变种。

@DATA

```
5.1, 3.5, 1.4, 0.2, Iris-setosa
7.0, 3.2, 4.7, 1.4, Iris-versicolor
6.3, 3.3, 6.0, 2.5, Iris-virginica
4.9, 3.0, 1.4, 0.2, Iris-setosa
6.4, 3.2, 4.5, 1.5, Iris-versicolor
5.8, 2.7, 5.1, 1.9, Iris-virginica
...
```

- 应用 Cobweb 于 Iris-cobweb2.arff 并以不同的 Acuity 值进行实验。哪一个 Acuity 值产生最小的误差率?通常该误差率应小于由 Iris-cobweb1.arff 得到的误差率。
- 比较使用 Iris-cobweb1.arff 和 Iris-cobweb2.arff 产生的簇的个数。
- 使用可视化工具显示簇及分配给各个簇的实例。

参考文献

- [1] Sneath and Sokal, *Numerical Taxonomy*, Freeman, San Fransisco, CA, 1973.
- [2] Ward, J.H., Hierarchical Grouping to Optimize an Objective Function, *J. Am. Staist. Assoc.*, 58, pp. 236-244, 1963.
- [3] M. Ester, H.-P. Kriegel, J. Sander, Xu X., A Density-based algorithm for discovering clusters in large spatial databases with noise, *Proc. 2nd Int. Conf. On Knowledge Discovery and Data Mining (KDD'96)*, Portland, OR, [4], pp. 226-31, 1996.
- [4] M. Ankerest, M.M. Breunig, H.-P. Kriegel, and J. Sander, OPTICS, *Proc. ACM SIGMOD'99*, Int. Conf. on Management of Data, Philadelphia, pp. 49-60, 1999.
- [5] George Karypis, Eui-Hong (Sam) Han, and Vipin Kumar, CHAMELEON: A hierarchical clustering algorithm using dynamic modeling, *IEEE_Computer*, 1999.
- [6] Sudipto Guha, Rajeev Rastogi and Kyuseok Shim, CURE: An efficient clustering algorithm for large databases, In: *Proc. of 1998 ACM-SIGMOD*, Int. Conf. on Management of Data, 1998.
- [7] G. Karypis, E.H. Han, and V. Kumar, CHAMELEON: A Hierarchical clustering algorithm using dynamic modeling, *IEEE Computer*, 32(8): 68-75, 1999.
- [8] G. Karypis and V. Kumar, hMETIS 1.5: A Hypergraph Partitioning Package, *Technical Report, Department of Computer Science*, University of Minnesota, 1998, Available on the WWW at URL <http://www.cs.umn.edu/~metis>.
- [9] G. Karypis and V. Kumar, METIS4.0: Unstructured graph partitioning and sparse matrix ordering system, *Technical Report, Department of Computer Science*, University of Minnesota, 1998, Available on the WWW at URL <http://www.cs.umn.edu/~metis>.
- [10] Y. Zhao and G. Karypis, Evaluation of hierarchical clustering algorithms for document datasets. In: *CIKM*, 2002.
- [11] Y. Zhao and G. Karypis, Criterion functions for document clustering: Experiments and analysis. *Technical Report TR #01-40, Department of Computer Science*, University of Minnesota, Minnaeapolia, MN, 2001, Available on the WWW at URL <http://www.cs.umn.edu/~karypis/publications>.
- [12] Winston and Albright, *Practical Management Science*, 2nd ed., Clustering algorithms divide data into meaningful or useful groups, called clusters, such that the *intra-cluster similarity is maximized and the inter-cluster similarity is minimized*.
- [13] T. Zhang, R. Ramakrishnan, and M. Living, Birch: An efficient data clustering algorithm for very large databases. In *SIGMOD*, QB, 1996.

第 12 章 多维数据可视化

12.1 引言

可视化是数据挖掘中非常重要的技巧,因为人类在处理可视化信息方面能力卓越。人类可以在瞬间从复杂的可视场景中提取出重要的特征。人类具有很好的可视化信息处理技巧,可以用能被人脑快速处理的形式来表示出复杂的信息。

可视化可应用于数据挖掘过程中的若干情形。它可以作为数据选择和准备工具,向领域专家提供从何处挖掘数据的线索。它也可以用终端用户(如领域专家)更易理解的方式展示数据挖掘结果。最近,它也直接用于通过交互式探索数据分析进行数据挖掘。

多维可视化技巧几乎在每个学科都有有用的应用,因为大多数学科都采用简单的模型来更好地显示复杂系统的系统行为。使用现代技术,计算机可用于使更复杂的模型可视化,并且对更复杂的问题提供简化的多维解决方案。

“可视化”并非新技术。例如,法国发现的洞穴壁画已经有两万年的历史。中国人在 12 世纪就创造了人类已知的最早的地图。但是,直到 19 世纪才出现最早的多维(信息)表示。John Snow 博士和 Charles Joseph Minard 给出了两个最好的例子。1854 年, Snow 博士标绘出了伦敦的霍乱死亡者。他用点标出死亡者的地点并用叉标出水泵的位置。他发现霍乱几乎全部发生在居住在百老大街(Broad Street)水泵附近(或饮用该水泵取出的水)的人群中。基于这一观察结果,他发现需要移走水泵,进而终止霍乱[1]。

Charles Joseph Minard 是一位法国道路和桥梁工程师及监察员,他在 150 年前创造了关于拿破仑征俄战争^①的最令人印象深刻的多维图形表示。有时候这称之为“历史上最好的统计图形”,以及“挑战历史学家之笔”的工作。Minard 绘制了一个连续图,描绘了拿破仑大军在灾难性的 1812 年征俄战役中的悲惨命运。使用笔墨, Minard 在二维纸张上捕捉了不少于六维的描述性数据(见图 12-1)。

Edward Tufte[1]是一位在理解数据的艺术和科学领域耕耘 30 余载的信息设计者,他极具说服力地描绘了 Minard 的图表。

图像中间最粗的线条表示拿破仑在 1812 年 6 月从波兰-俄国边界近尼尔曼河入侵俄国时的军队力量,达 422 000 人之多。随着军队的进入,线条的粗度因军力在进军莫斯科的途中随着人员损失而不断变细。在军队抵达莫斯科时(图中的最右边)减少到了 100 000 人,只有最初的 1/4。下面的黑线代表了拿破仑军队的撤退以及俄国严寒的灾难性效果。撤退线条和图像下面的日期以及温度相联系。等军队返回到波兰时,由于严寒导致军队减少到 10 000 人。除了主力军队, Minard 还描绘了辅助军队保护主力军队两翼的行动。

Minard 的图表是数据表示脱离二维限制的鼎盛之作。他传递了关于世界的一个核心现实:事物是多维的。Minard 捕捉并绘制了 6 个变量:军力(1);军队在二维平面上的位置(2, 3);军队移动的方向(4);从莫斯科撤退过程中各天的温度(5, 6)。

① 原文是 human folly, 实际是指拿破仑征俄战争。——译者注

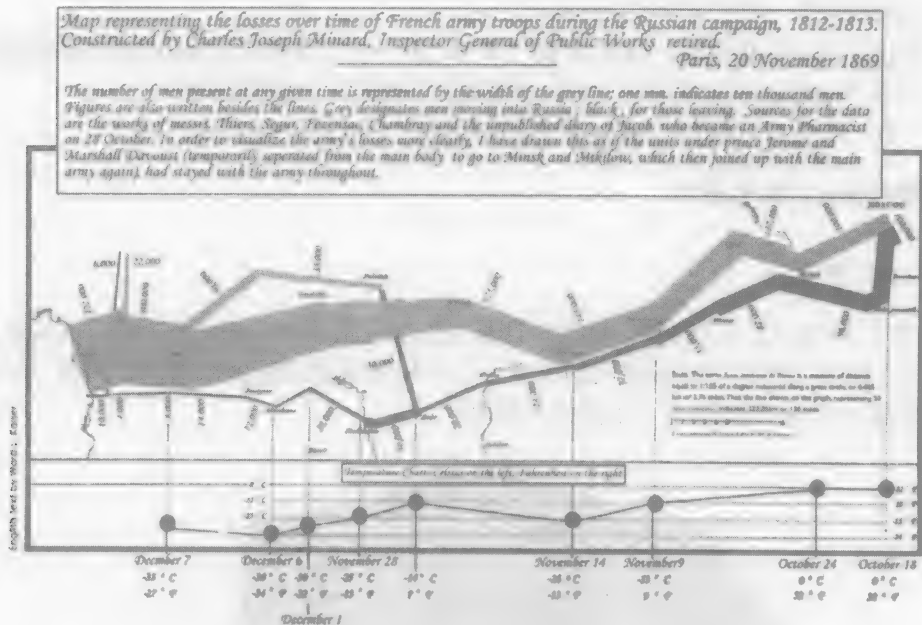


图 12-1 Minard 的图表

现在，计算机科学为我们提供了以可视化方式表示信息的革命性工具。其威力通过今天视频游戏的巨大吸引力表露无遗。虽然经常以混乱和毁灭示人，视频游戏中展现的 3D 影像的惊人诱惑力将视频游戏引领到一个新的视觉世界。它还具有从多维显示数据，呈现信息的能力。

究竟什么是可视化？新韦伯英语词典[1989, P. 1100]对可视化的定义是“一个可视或被可视的过程；一个心理图像”。而什么又是“被可视”？蓝登书屋英语词典[1987, P. 2127]对可视化的定义是“回忆或形成一个心理映像或图片；使可视或可见；对事物形成一个心理映像；对思想或想象进行感知”。这些定义暗示着可视化是某些现象的一个图像。“计算机图像和虚拟现实词典”中一个更为恰当的定义是，可视化是“将数据表示为视觉图像的过程”[2]。所基于的数据可能代表具体的对象（如房子或汽车），或代表抽象的对象（如利润、销售或成本）。如果数据是抽象的，那么必须对其创建直观模拟。典型直观模拟是饼图或曲线图。

可视化的目标并非替代具有坚实基础的量化分析，而是要加强量化分析。可视化可以：

- 利用人类的视觉系统从数据中提取信息。
- 对复杂数据集提供概览。
- 标识数据中的结构、模式、趋势、异常和联系。
- 协助标识出“感兴趣”的区域。

换句话说，可视化允许决策者使用他们天生的空间/时间能力，决定在哪些地方继续探索。这意味着，如果可视化被恰当利用，可使决策者从数据中发现信息。

可视化技术分为三大类：科学可视化、数据/信息可视化和虚拟现实。

- 科学可视化，顾名思义，处理科学或工程计算或实验中产生的数据到图像的转换，如滑过机翼的气流。
- 数据/信息可视化用于将非空间或行为数据转化为可视图像，用以表示问题空间的一

个模拟或象征,如投资档案分析。

- 在商业信息可视化领域,虚拟现实(VR)只是一个三维的计算机生成的模拟系统,用来对用户行为进行实时展现。VR也称为人工现实以及虚拟环境[3]。
- 在下一节,我们讨论信息可视化设计者发明的一个可视化表示的例子。

12.2 多维可视化的图表表示

12.2.1 kiviati 图

kiviati 图(参见图 12-2 和图 12-3)已在计算机性能评估领域应用多年[Kolence and Kiviati, 1973]。kiviati 图可以描述多元数据间的关系。每个度量值显示其坐标轴上。例如,如果有五个独立的变量,kiviati 图将有五个不同的坐标轴。

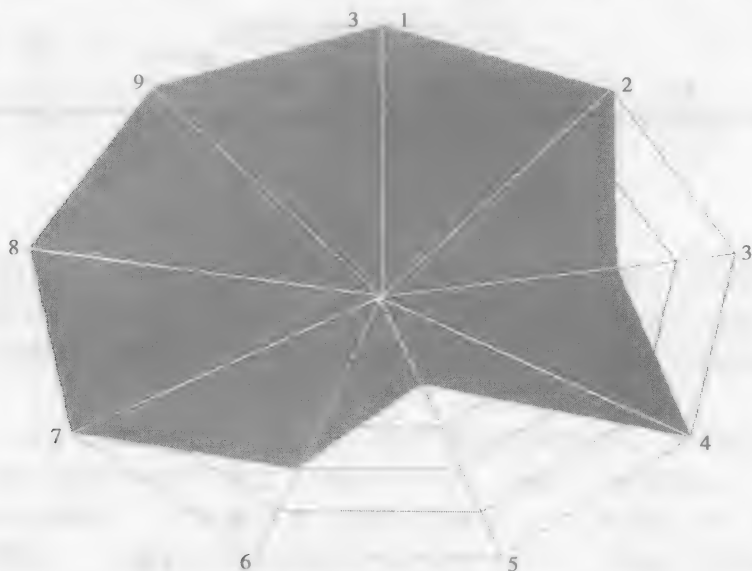


图 12-2 kiviati 图(或星状图)

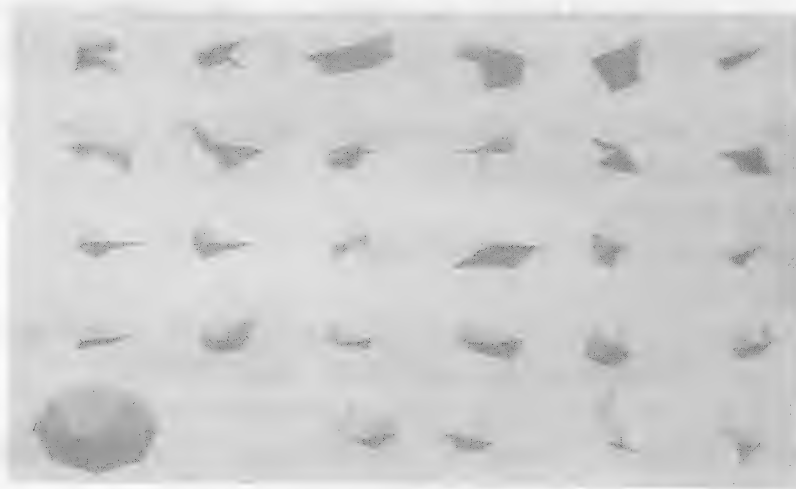


图 12-3 表示不同对象的 kiviati 图集合

将对感兴趣的实体的度量值绘制在相应的坐标轴上。然后连接数据点。最后形成的模式即为信息的可视化。为了比较不同实体，可对其模式进行比较。这类图也称为雷达图、星状图、蜘蛛图或星云图。

12.2.2 平行坐标系

平行坐标系(参见图 12-4)是另一种长期使用的多元数据可视化技术。最近，该方法开始应用于数据挖掘领域。如图 12-4 所示，和 Kiviat 图一样，关于每个实体的度量被绘制在其相应的坐标轴上。在这种方法中，(表现)模式是线条而非多边形，使得查找不同实体之间的相似模式变得非常直接。

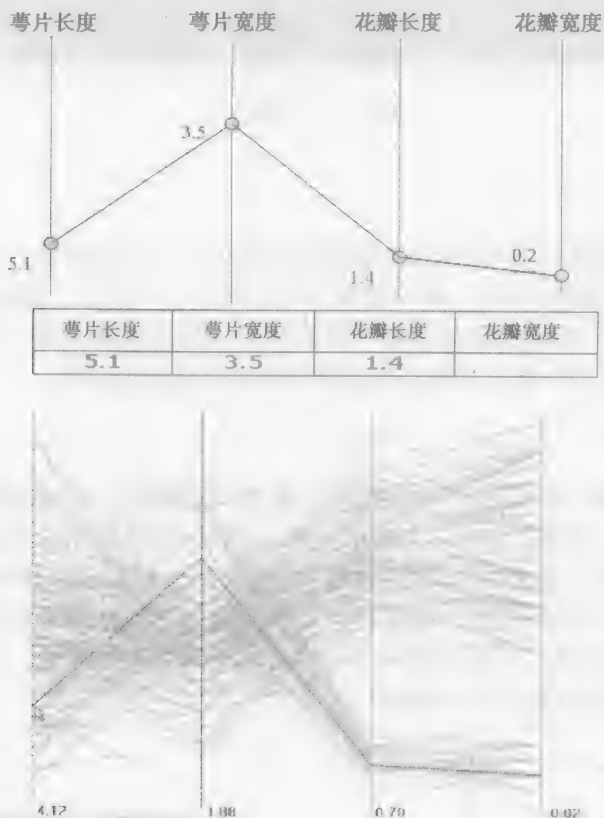


图 12-4 平行坐标系

12.2.3 3D 散点图

3D 散点图(见图 12-5)是统计工具包中常见的 2D 散点图的扩展。在这种方法中，可对一个实体的最多四种不同度量进行表示，包括每个坐标轴(x 、 y 和 z)以及颜色。但是，该方法的最大的缺点是很难精确确定每个度量值的位置。大多数情况下，需要增加参考数据，如对相应的坐标值增加来自其真实值的线条并增加色例(colour map)。否则，对散点图的解释会产生问题。

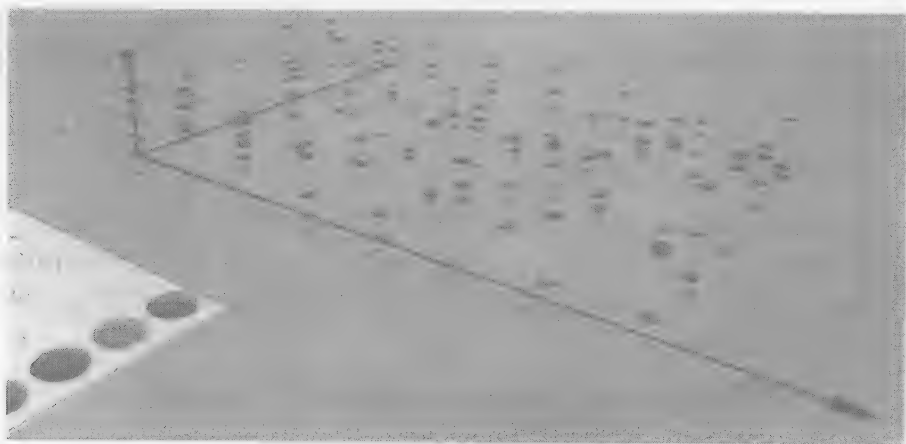


图 12-5 3D 散点图

12.2.4 3D 曲线图

三维曲线图(参见图 12-6)是典型二维曲线图的扩展。它可表示一个感兴趣的实体的最多四种不同度量。在这种方法中,曲线穿越三维空间(x 、 y 和 z)。随着第 4 个度量的变化,线条的颜色随之变化。由于每次只绘制一个实体,因此其结果相较于 3D 散点图更便于解释。然而,和 3D 散点图一样,其表示也需要增加参考数据和色例。

12.2.5 体积透视图

体积透视图(volume rendering)(见图 12-7)要求(数据源是)3D 数据集。该方法已被作为科学可视化技术使用。然而,Silicon Graphics 的 Becker[4]最近用其表示存储于关系数据集中的数据。在其工作中,他将聚集后的数据(教育水平、职业和工作时间)绘制在 3D 空间中,并为因变量(收入)赋予颜色。可视化的不透明度基于该区域包含的观测数量。他还将一个额外变量(年龄)赋予一个外部滑块(slider)作为可视查询的基础。基于该工作,体积透视图也许会成为其他多元表示的合理替代。

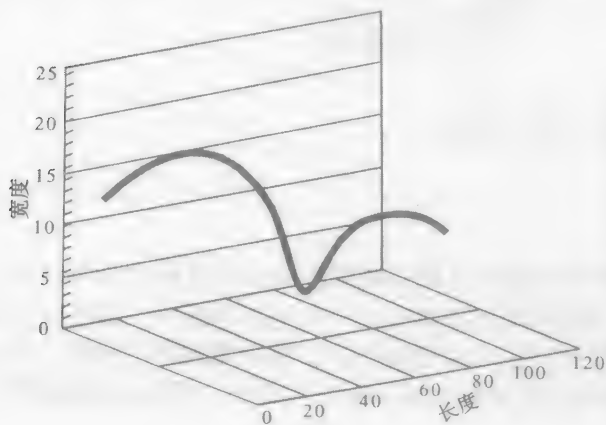


图 12-6 3D 曲线图

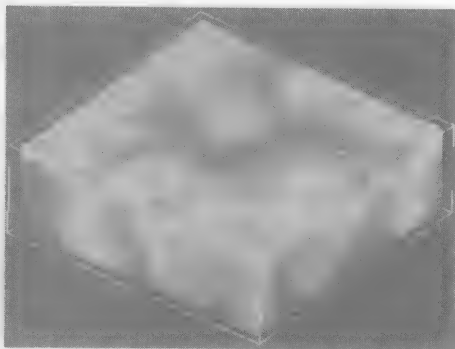


图 12-7 体积透视图

12.2.6 房图

房图(floors and walls)(见图 12-8)是一种房间象征形式。在这种表示中,信息被赋值于各种商业图形并显示在房间的墙壁或地板上。这种表示允许在相对狭小的空间表示大量信息,并且它为决策支持者提供他们熟悉的图形,如饼图、柱状图、曲线图以及其他典型的商业图表方法。这种表示支持探索性以及验证性决策任务。在这类任务中,决策制定者在(立体图形成的)景观中导航来发现新的假设或对已有假设进行验证。

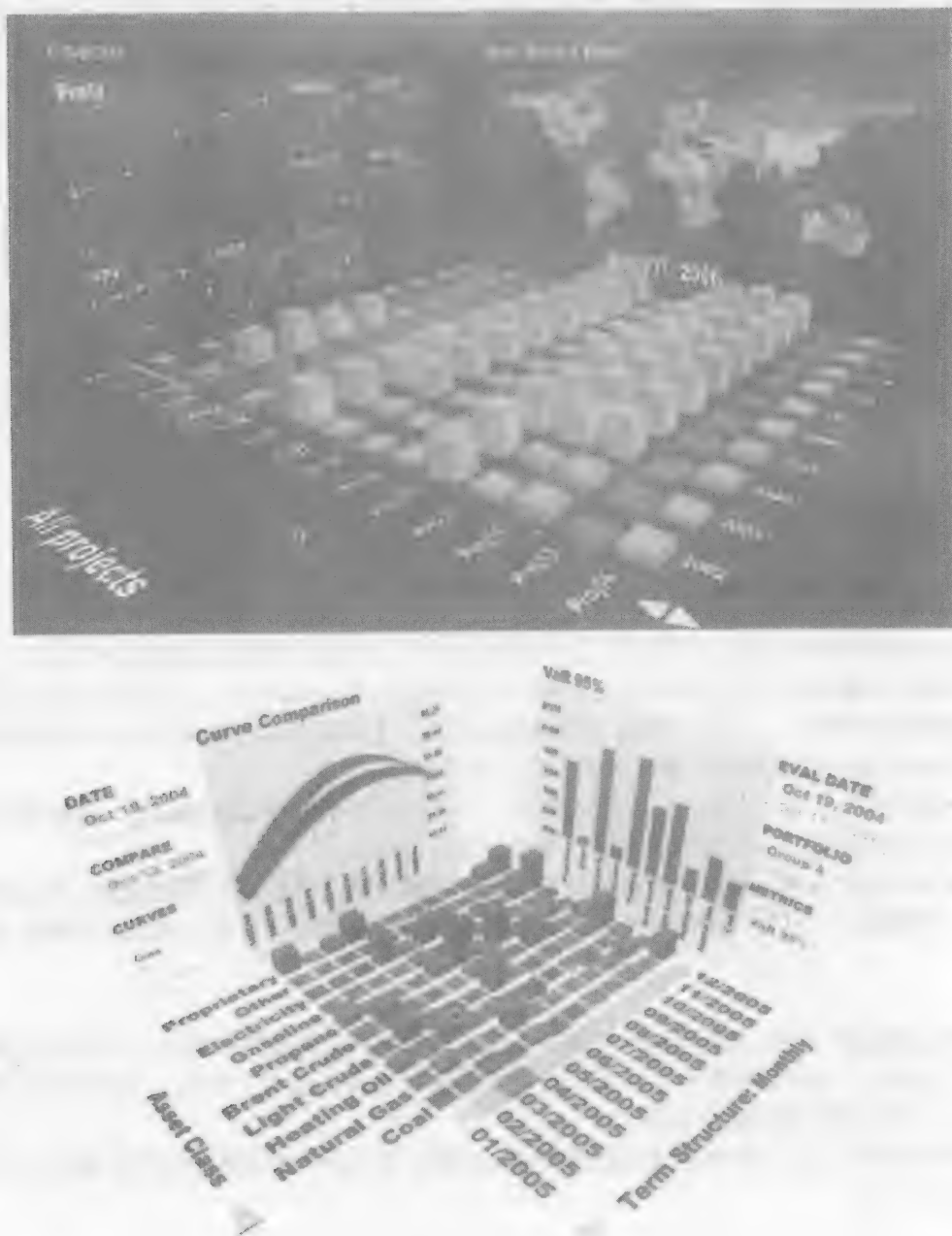


图 12-8 房图

12.2.7 Chernoff 脸图

1973年, Herman Chernoff 引入了一种可视化技术——Chernoff 脸图, 以表示多维数据中的趋势。他的 Chernoff 脸图(参见图 12-9)是非常有效的, 因为它将数据和脸部特征相联系, 而脸部特征正是我们用于区分人的重要特征。不同的数据维被映射到不同的脸部特征, 如脸的宽度、耳朵的高度、耳朵的半径、嘴的长度或曲率、鼻子的长度等。图 12-9 是 Chernoff 脸图(Chernoff face)的一个例子。它用脸部特征来表示数据值中的趋势, 而非数值本身。虽然这是一个明显的局限, 但是数据中的趋势信息可以用来帮助决定哪部分数据是有意义的。

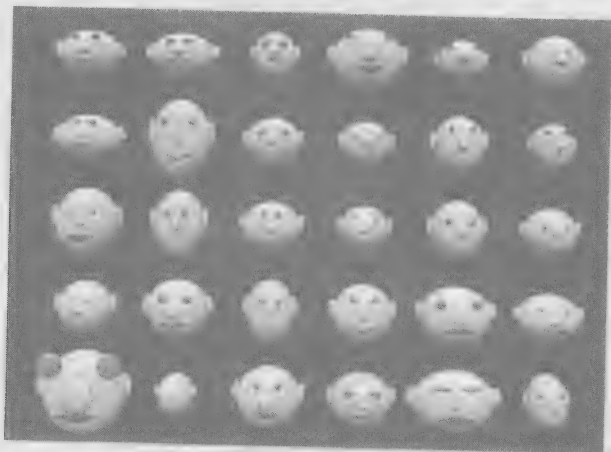


图 12-9 Chernoff 脸图

12.3 可视化数据挖掘

许多现代数据可视化工具结合了强大的可视化显示和易于操作的数据选择和显示控制。这些功能允许领域专家交互式地探索数据。这些功能是如此的高效, 以至于在不使用自动数据挖掘算法的情况下, 也可以发现有趣的数据模式。这类数据挖掘有时称为可视化数据挖掘(visual data mining)。好的可视化数据挖掘工具具有以下功能:

- 在可视化背景下的交互式导航的能力, 允许在显示的数据上进行缩放、旋转和扫描。
- 交互式控制显示数据的显示格式和可视属性的能力。
- 交互式控制显示数据的粒度的功能, 允许领域专家从高层视角观测数据或下钻到特定的数据集。这使得领域专家可以分析所显示的信息的总体, 或专注于细节和异常。

动画

最近的数据可视化工具还包括数据动画技术。常见的可视化技术顺应了人类处理复杂视觉信息的能力。动画技术顺应了人类发现可视信息中动作的能力。为此, 动画是分析数据, 尤其是分析时态数据的有力工具。

数据动画技术的工作原理通常是选择准则变量并用动画工具来显示这些变量的不同值的数据记录行为。

参考文献

- [1] E.R. Tufte, *The Visual Display of Quantitative Information*, Graphics, Press, Cheshire, Conn., 1983.
- [2] R. Latham, *The Dictionary of Computer Graphics and Virtual Reality*, 2nd ed., Springer-Verlag, New York, 1995.
- [3] C.E. Loeffler and T. Anderson (Eds.), *The Virtual Reality Casebook*, Van Nostrand Reinhold, New York, 1994.
- [4] B. Becker, Research Report: Volume rendering for relational data, in *Proceedings Information Visualization*, John Dill and Nahum Gershonn (Eds.), IEEE-CS Press, Los Alamos, CA, 124, pp. 87-90, 1997.
- [5] H. Chernoff, The use of faces to represent points in k -dimensional space graphically, *Journal of the American Statistical Association*, 68, pp. 361-368, 1973.

附录 A SVM 公式：完全可分的线性分类器

分离超平面由下式给出：

$$\mathbf{w}^T \mathbf{w} - \gamma = 0 \quad \mathbf{w} \in R^n, \quad \gamma \in R$$

对应的决策函数是：

$$f(x) = \text{sign}(\mathbf{w}^T \mathbf{x} - \gamma)$$

为了建立最佳超平面，我们对下面的优化问题求解：

$$\min_{\mathbf{w}, \gamma} \frac{1}{2} \|\mathbf{w}\|^2$$

满足： $(\mathbf{w}^T \mathbf{x}_i - \gamma) \geq +1, i=1, 2, \dots, m_1$ ，对于属于 +1 类的样本

$(\mathbf{w}^T \mathbf{x}_j - \gamma) \leq -1, j=1, 2, \dots, m_2$ ，对于属于 -1 类的样本 (A-1)

这些不等式公式可以表达为：

$$D_{ii}(\mathbf{w}^T \mathbf{x}_i - \gamma) \geq 1, i=1, 2, \dots, m_1$$

$$D_{jj}(\mathbf{w}^T \mathbf{x}_j - \gamma) \geq 1, j=1, 2, \dots, m_2$$

其中，对所有属于 +1 类的数据有 $D_{ii} = +1$ ，对其他类的有 $D_{jj} = -1$ 。上述两个不等式现在可以组合为一个不等式：

$$D_{kk}(\mathbf{w}^T \mathbf{x}_k - \gamma) \geq 1, k=1, 2, \dots, m$$

其中 $m = m_1 + m_2$ 。

对上述问题求解的一种方法是求解拉格朗日对偶问题：

$$\max_{\mathbf{u} \geq 0} [\min_{\mathbf{w}, \gamma} L(\mathbf{w}, \gamma, \mathbf{u})] \quad (\text{A-2})$$

其中

$$L(\mathbf{w}, \gamma, \mathbf{u}) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{k=1}^m u_k [D_{kk}(\mathbf{w}^T \mathbf{x}_k - \gamma) - 1] \quad (\text{A-3})$$

对于给定的 \mathbf{u} ，我们寻找最小化 $L(\mathbf{w}, \gamma, \mathbf{u})$ 的 \mathbf{w} 和 γ 值，并代入式(A-3)。为了取最小值，我们有：

$$\frac{\partial}{\partial \gamma} L(\mathbf{w}, \gamma, \mathbf{u}) = 0 \quad \text{和} \quad \frac{\partial}{\partial \mathbf{w}} L(\mathbf{w}, \gamma, \mathbf{u}) = 0$$

这导致

$$\sum_{k=1}^m u_k D_{kk} = 0 \quad (\text{A-4})$$

和

$$\mathbf{w} = \sum_{k=1}^m u_k D_{kk} \mathbf{x}_k \quad (\text{A-5})$$

将式(A-5)代入式(A-3)，我们得到

$$\begin{aligned} L(\mathbf{w}, \gamma, \mathbf{u}) &= \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^m u_i [D_{ii}(\mathbf{w}^T \mathbf{x}_i - \gamma) - 1] \\ &= \frac{1}{2} (\mathbf{w}^T \mathbf{w}) - \sum_{i=1}^m u_i [D_{ii}(\mathbf{w}^T \mathbf{x}_i - \gamma) - 1] \end{aligned}$$

$$= \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m u_i u_j D_{ii} D_{jj} (x_i^T x_j) - \sum_{i=1}^m u_i D_{ii} (w^T x_i) + \gamma \sum_{i=1}^m u_i D_{ii} + \sum_{i=1}^m u_i$$

因为根据式(A-4)有 $\sum_{i=1}^m u_i D_{ii} = 0$, 于是

$$\begin{aligned} L(w, \gamma, u) &= \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m u_i u_j D_{ii} D_{jj} (x_i^T x_j) - \sum_{i=1}^m \sum_{j=1}^m u_i u_j D_{jj} D_{ii} (x_j^T x_i) + \sum_{i=1}^m u_i \\ &= \sum_{i=1}^m u_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m u_i u_j D_{ii} D_{jj} (x_i^T x_j) \end{aligned}$$

现在, 拉格朗日函数是仅仅关于 u 的函数, 我们将这个函数表示为:

$$L_D(u) = \sum_{i=1}^m u_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m u_i u_j D_{ii} D_{jj} (x_i^T x_j)$$

我们知道 u 必须满足式(A-4)给定的条件。

现在优化问题归结为:

$$\text{Max } L_D(u) = \sum_{i=1}^m u_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m u_i u_j D_{ii} D_{jj} (x_i^T x_j)$$

满足

$$\sum_{i=1}^m u_i D_{ii} = 0, u_i \geq 0, i = 1, 2, \dots, m \quad (\text{A-6})$$

上述二次规划问题可得到 u_i s, 而一旦得到这个值, 决策函数可由下式给出:

$$f(x) = \text{sign}(w^T x - \gamma) = \text{sign}\left(\sum_{i=1}^m u_i x_i^T x - \gamma\right)$$

γ 的值可通过在边缘上取一个 $u_i \neq 0$ 的点而轻易求得。

线性 SVM 的矩阵形式

$$\underset{w, \gamma}{\text{minimize}} \frac{1}{2} \|w\|^2$$

满足

$$D_{kk}(w^T x_k - \gamma) \geq 1, k = 1, 2, \dots, m$$

还可以表达为:

$$\underset{w, \gamma}{\text{minimize}} \frac{1}{2} (w^T w)$$

满足

$$D(Aw - e\gamma) \geq e \quad (\text{A-7})$$

其中 e 是 1 的 $m \times 1$ 向量, 即

$$e = [1, 1, 1, \dots, 1, 1]^T$$

$$L(w, \gamma, u) = \frac{1}{2} (w^T w) - u^T [D(Aw - e\gamma) - e]$$

$$\frac{\partial}{\partial \gamma} L(w, \gamma, u) = 0 \Rightarrow e^T D^T u = e^T D u = 0 \quad (\text{A-8})$$

$$\frac{\partial}{\partial w} L(w, \gamma, u) = 0 \Rightarrow w - A^T D u = 0 \quad (\text{A-9})$$

因此有

$$\begin{aligned} L_D(u) &= \frac{1}{2} (A^T D u)^T A^T D u - u^T [D(AA^T D u - e\gamma) - e] \\ &= \frac{1}{2} (u^T D A A^T D u) - u^T D A A^T D u + \gamma u^T D e + u^T e \end{aligned} \quad (\text{A-10})$$

注意, 上述表达式中的每个项都是标量, 并且第三项 $\gamma u^T D e$ 与 $\gamma e^T D u$ 相同。因为由式(A-8)有 $e^T D u = 0$, 式(A-10)简化为:

$$L_D(u) = u^T e - \frac{1}{2} u^T D A A^T D u$$

因此, 对偶优化问题为:

$$\text{最大化} \quad L_D(u) = u^T e - \frac{1}{2} u^T D A A^T D u$$

$$\text{满足} \quad e^T D u = 0, \quad u \geq 0$$

决策函数 $f(x)$ 由下式给出:

$$f(x) = \text{sign}(w^T x - \gamma) = \text{sign}[(A^T D u)^T x - \gamma] = \text{sign}(u^T D A x - \gamma)$$

完全可分的非线性分类器

$w^T \phi(x) - \gamma = 0$, $w \in F$ (特征空间维数决定于非线性函数 ϕ), $\gamma \in R$, 对应的决策函数是:

$$f(x) = \text{sign}[w^T \phi(x) - \gamma]$$

为了构建最佳超平面, 对下面的优化问题求解:

$$\text{minimize}_{w, \gamma} \frac{1}{2} \|w\|^2$$

满足: $(w^T \phi(x_i) - \gamma) \geq +1, i = 1, 2, \dots, m_1$, 对于属于 +1 类的样本

$(w^T \phi(x_j) - \gamma) \leq -1, j = 1, 2, \dots, m_2$, 对于属于 -1 类的样本 (A-11)

这些不等式可以表达为:

$$D_{ii}(w^T \phi(x_i) - \gamma) \geq 1, i = 1, 2, \dots, m_1$$

$$D_{jj}(w^T \phi(x_j) - \gamma) \geq 1, j = 1, 2, \dots, m_2$$

其中, 对所有属于 +1 类的数据有 $D_{ii} = +1$, 对其他类的有 $D_{jj} = -1$ 。上述两个不等式现在可以组合为一个不等式:

$$D_{kk}(w^T \phi(x_k) - \gamma) \geq 1, k = 1, 2, \dots, m$$

其中 $m = m_1 + m_2$ 。对上述问题求解的一种方法是求解拉格朗日对偶问题:

$$\max_{u \geq 0} (\min_{w, \gamma} L(w, \gamma, u)) \quad (\text{A-12})$$

其中

$$L(w, \gamma, u) = \frac{1}{2} \|w\|^2 - \sum_{k=1}^m u_k [D_{kk}(w^T \phi(x_k) - \gamma) - 1] \quad (\text{A-13})$$

对于给定的 u , 我们寻找最小化 $L(w, \gamma, u)$ 的 w 和 γ 值, 并代入式(A-13)。为了取最小值, 我们有:

$$\frac{\partial}{\partial \gamma} L(w, \gamma, u) = 0 \text{ 和 } \frac{\partial}{\partial w} L(w, \gamma, u) = 0$$

这导致:

$$\sum_{k=1}^m u_k D_{kk} = 0 \quad (\text{A-14})$$

和

$$w = \sum_{k=1}^m u_k D_{kk} \phi(x_k) \quad (\text{A-15})$$

将式(A-15)代入式(A-13), 得到

$$\begin{aligned} L(\mathbf{w}, \gamma, \mathbf{u}) &= \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^m u_i [D_{ii}(\mathbf{w}^T \phi(\mathbf{x}_i) - \gamma) - 1] \\ &= \frac{1}{2} (\mathbf{w}^T \mathbf{w}) - \sum_{i=1}^m u_i [D_{ii}(\mathbf{w}^T \phi(\mathbf{x}_i) - \gamma) - 1] \\ &= \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m u_i u_j D_{ii} D_{jj} \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) - \sum_{i=1}^m u_i D_{ii} (\mathbf{w}^T \phi(\mathbf{x}_i)) \\ &\quad + \gamma \sum_{i=1}^m u_i D_{ii} + \sum_{i=1}^m u_i \end{aligned}$$

因为根据式(A-14), 有 $\sum_{i=1}^m u_i D_{ii} = 0$, 于是

$$\begin{aligned} L(\mathbf{w}, \gamma, \mathbf{u}) &= \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m u_i u_j D_{ii} D_{jj} [\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)] - \sum_{i=1}^m \sum_{j=1}^m u_i u_j D_{ii} D_{jj} [\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)] \\ &\quad + \sum_{i=1}^m u_i = \sum_{i=1}^m u_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m u_i u_j D_{ii} D_{jj} [\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)] \end{aligned}$$

现在拉格朗日函数是仅仅关于 \mathbf{u} 的函数, 我们将这个函数表示为:

$$L_D(\mathbf{u}) = \sum_{i=1}^m u_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m u_i u_j D_{ii} D_{jj} [\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)]$$

我们知道 \mathbf{u} 必须满足式(A-14)给定的条件。现在, 优化问题归结为:

$$\text{Max } L_D(\mathbf{u}) = \sum_{i=1}^m u_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m u_i u_j D_{ii} D_{jj} [\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)]$$

满足

$$\sum_{i=1}^m u_i D_{ii} = 0, u_i \geq 0, i = 1, 2, \dots, m \quad (\text{A-16})$$

上述二次规划问题可得到 $u_i s$, 而一旦得到这个值, 决策函数可由下面的公式给出:

$$f(\mathbf{x}) = \text{sign}(\mathbf{w}^T \phi(\mathbf{x}) - \gamma) = \text{sign} \left[\sum_{i=1}^m u_i \phi(\mathbf{x}_i)^T \phi(\mathbf{x}) - \gamma \right]$$

γ 的值可通过在边缘上取一个 $u_i \neq 0$ 的点而求得。

附录 B 图划分的矩阵形式

假设有 5 个点, 并且其相似度矩阵 W 为:

$$W = \begin{pmatrix} 1 & s_{12} & s_{13} & s_{14} & s_{15} \\ s_{21} & 1 & s_{23} & s_{24} & s_{25} \\ s_{31} & s_{32} & 1 & s_{34} & s_{35} \\ s_{41} & s_{42} & s_{43} & 1 & s_{45} \\ s_{51} & s_{52} & s_{53} & s_{54} & 1 \end{pmatrix}$$

注意, 这里 $s_{ij} = s_{ji}$ 。假设行的总和(列的总和)表示为 D_i :

$$D_1 = 1 + s_{12} + s_{13} + s_{14} + s_{15}$$

$$D_2 = 1 + s_{21} + s_{23} + s_{24} + s_{25}$$

$$\vdots$$

$$D_5 = 1 + s_{51} + s_{52} + s_{53} + s_{54}$$

令

$$D = \begin{pmatrix} D_1 & 0 & 0 & 0 & 0 \\ 0 & D_2 & 0 & 0 & 0 \\ 0 & 0 & D_3 & 0 & 0 \\ 0 & 0 & 0 & D_4 & 0 \\ 0 & 0 & 0 & 0 & D_5 \end{pmatrix}$$

现在:

$D - W$

$$= \begin{pmatrix} s_{12} + s_{13} + s_{14} + s_{15} & -s_{12} & -s_{13} & -s_{14} & -s_{15} \\ -s_{21} & s_{21} + s_{23} + s_{24} + s_{25} & -s_{23} & -s_{24} & -s_{25} \\ -s_{31} & -s_{32} & s_{31} + s_{32} + s_{34} + s_{35} & -s_{34} & -s_{35} \\ -s_{41} & -s_{42} & -s_{43} & s_{41} + s_{42} + s_{43} + s_{45} & -s_{45} \\ -s_{51} & -s_{52} & -s_{53} & -s_{54} & s_{51} + s_{52} + s_{53} + s_{54} \end{pmatrix}$$

假设点 1、3、5 形成组 A , 点 2、4 形成另一个组 B , 那么 $X = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}$ 是一个表示组 A 的二

值(binary)向量。

现在,

$$X^T W = [1 \ 0 \ 1 \ 0 \ 1] \begin{pmatrix} 1 & s_{12} & s_{13} & s_{14} & s_{15} \\ s_{21} & 1 & s_{23} & s_{24} & s_{25} \\ s_{31} & s_{32} & 1 & s_{34} & s_{35} \\ s_{41} & s_{42} & s_{43} & 1 & s_{45} \\ s_{51} & s_{52} & s_{53} & s_{54} & 1 \end{pmatrix}$$

$$= [(1 + s_{31} + s_{51})0(s_{13} + 1 + s_{53})0(s_{15} + s_{35} + 1)]$$

$$X^T W X = [(1 + s_{31} + s_{51})0(s_{13} + 1 + s_{53})0(s_{15} + s_{35} + 1)] \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}$$

= 组群 A 中元素间的权重的总和

如果我们将点 1, 3, 5 分到一组, 则相应的子矩阵如下:

$$W_A = \begin{pmatrix} s_{11} & s_{13} & s_{15} \\ s_{31} & s_{33} & s_{35} \\ s_{51} & s_{53} & s_{55} \end{pmatrix} = \begin{pmatrix} 1 & s_{13} & s_{15} \\ s_{31} & 1 & s_{35} \\ s_{51} & s_{53} & 1 \end{pmatrix}$$

可以看到, W_A 中的元素总和为 $X^T W X$ 。类似地:

$$W_B = \begin{pmatrix} s_{22} & s_{24} \\ s_{42} & s_{44} \end{pmatrix} = \begin{pmatrix} 1 & s_{24} \\ s_{42} & 1 \end{pmatrix}$$

可以看到 $Y^T W X = 1 + s_{24} + 1 + s_{42} = W_B$ 中的元素总和。

现在我们对 $X^T(D - W)X$ 求解。

$$X^T(D - W) = [1 \ 0 \ 1 \ 0 \ 1]$$

$$\begin{pmatrix} s_{12} + s_{13} + s_{14} + s_{15} & -s_{12} & -s_{13} & -s_{14} & -s_{15} \\ -s_{21} & s_{21} + s_{23} + s_{24} + s_{25} & -s_{23} & -s_{24} & -s_{25} \\ -s_{31} & -s_{32} & s_{31} + s_{32} + s_{34} + s_{35} & -s_{34} & -s_{35} \\ -s_{41} & -s_{42} & -s_{43} & s_{41} + s_{42} + s_{43} + s_{45} & -s_{45} \\ -s_{51} & -s_{52} & -s_{53} & -s_{54} & s_{51} + s_{52} + s_{53} + s_{54} \end{pmatrix}$$

$$= [(s_{12} + s_{14}) - (s_{12} + s_{32} - s_{52})(s_{32} + s_{34}) - (s_{14} + s_{34} + s_{54})(s_{52} + s_{54})]$$

$$X^T(D - W)X = [(s_{12} + s_{14}) - (s_{12} - s_{32} - s_{52})(s_{32} + s_{34}) - (s_{14} + s_{34} + s_{54})(s_{52} + s_{54})] \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}$$

$$= (s_{12} + s_{14}) + (s_{32} + s_{34}) + (s_{52} + s_{54})$$

= 组 A 和组 B 间边的权重的总和。

这一点可以通过下面表示组 A 中节点和 B 中节点之间权重的子矩阵来看到。 $Y^T(D - W)Y$ 即为该总和, 因为 $s_{ij} = s_{ji}$ 。

$$(2 \ 4) \begin{pmatrix} 1 \\ 3 \\ 5 \end{pmatrix} \begin{pmatrix} s_{12} & s_{14} \\ s_{32} & s_{34} \\ s_{52} & s_{54} \end{pmatrix}$$